LECTURE NOTES
The Logical Foundations of Computer Science and Mathematics

Maintainer: Anthony A. Aaby

An Open Source Text
DRAFT Version: α 0.3
Last Edited: August 7, 2006

Walla Walla College
204 S. College Ave.
College Place, WA 99324
E-mail: aabyan@wwc.edu

The author solicits collaboration with others on the elaboration and extension of the material in this text. Users are invited to suggest and contribute material for inclusion in future versions provided it is offered under compatible copyright provisions. The most current version of this text and LaTeXsource is available at `http://www.cs.wwc.edu/~aabyan/Logic/index.html`.

WWC Computer Science Students

Contents

List of Figures

Chapter 1

Before you begin ...

> This is a work in progress. Feedback to aabyan@wwc.edu is appreciated.

This is a collection of formal and informal notes, ideas, and essays on ...

Goals for this work...

- Provide a useful reference of the central ideas of formal logic
- Concepts from Logic that are important or useful for computer science and software engineering.

As far as I know, Chapter **??** is an orginal work.

Chapter 2

Sets, Relations, and Functions

2.1 Informal Set Theory

> (Set theory is) The finest product of mathematical genius and one of the supreme achievements of purely intellectual human activity. - *David Hilbert*

Bag A *bag* is an unordered collection of elements. It is also called a *multiset* and may include duplicates.

Set A *set* is an unordered collection of *distinct* elements selected from a *domain of discourse* or the *universe of values*, U. S, X, A, B, ... are sets. a, b, x, y, ... are elements.

Set definition/description Sets may be defined by *extension*: specification by explicit listing of members $A = \{x_0, ..., x_n\}$. A set of one element is called a *singleton set*. Sets may be defined by *intension/comprehension*: specification by a membership condition or rule for inclusion in the set. $A = \{x|P(x)\} = \{x : P(x)\}$. $P(x)$ is usually a logical expression. As no restrictions are placed on the condition or rule this method is called the *unrestricted principle of comprehension* or *abstraction*.

Some Sets

$\emptyset = \{x|x \neq x\}$ The *empty set* is the set that has no elements.
$\mathcal{U}$ The set of all the elements in the universe of discourse.
$\mathbb{N} = \{0, 1, 2, ...\}$ The set of natural numbers.
$\mathbb{Z} = \{... -1, 0, 1, ...\}$ The set of integers.

$\mathbb{Q}^+ = \{p/q \mid p, q \in \mathbb{N}$ and $q \neq 0\}$ The set of positive rationals.
$\mathbf{D}_n = \{d_0, d_1, ..., d_{n-1}\}$ some n in $\mathbb{N}$ called the base. The d_i are arbitrary symbols for digits.
$\mathbb{R}_n^+ = \{r \mid r = ...dd.dd...$; the d are posibly different elements of $\mathbf{D}_n$. These are the positive real numbers represented in base n.

With just the empty set and set definition we can create a sequence of sets that correspond to the natural numbers.

$$\begin{array}{lllll} \mathbb{N} = \{ & 0 & 1 & 2 & ...\} \\ & \emptyset & \{\emptyset\} & \{\emptyset, \{\emptyset\}\} & ... \end{array}$$

With the empty set and the power set construction we can create the set equivalent of the cosmological big bang:

$$\begin{array}{lllll} 0, & 1, & 2, & 4, & ... \\ \emptyset, & \{\emptyset\}, & \{\emptyset, \{\emptyset\}\}, & \{\emptyset, \{\emptyset\}, \{\{\emptyset\}\}, \{\emptyset, \{\emptyset\}\}\}, & ... \end{array}$$

Set operations

Membership: $x \in S$ x is an element (or member) of S.

Set Union - $A \cup B = \{x \mid x \in A$ **or** $x \in B\}$ The union of two sets is the set consisting of the elements of both sets.

Set Intersection - $A \cap B = \{x \mid x \in A$ **and** $x \in B\}$ The intersection of two sets is the set of the elements that they have in common.

Disjoint Sets - $A \cap B = \emptyset$ Two sets are disjoint if they have no elements in common.

Set difference - $A \sim B = A - B = A \backslash B = \{x | x \in A$ **and** $x \notin B\}$ The set difference is the set of elements found in the first set but not found in the second set.

Set complement - $A^c = \overline{A} = \sim A = \{x \mid x \notin A\} = \mathcal{U} \backslash \mathcal{A}$ The complement of a set A is a set consisting of those elements not found in the set A.

Cross product - $A \times B = \{(a, b) | a \in A$ **and** $b \in B\}$ The cross product of a pair of sets is the set of ordered pairs of elements from each set. *Note:* sets are unordered while pairs are ordered thus $\{a, b\} = \{b, a\}$ while the tuple $(a, b) \neq (b, a)$.

Subset: $A \subseteq B$ **if** $x \in A$ **then** $x \in B$ A set A is a subset of a set B if every element of A is an element of B.

Proper subset - $A \subset B = \{x |$ **if** $x \in A$ **then** $x \in B$ **and** $A \neq B\}$ A subset A of a set B is a proper subset if the sets are not equal.

Some theorems

- Theorem: $A = B$ iff $A \subseteq B$ and $B \subseteq A$.
- Theorem: $\emptyset \subset A$.
- Theorem: $x \notin x$ but $x \in \{x\}$.
- Theorem: $\emptyset \subset \wp(X)$ and $X \subset \wp(X)$
- Theorem: Idempotent - $A \cap A = A \cup A = A$
- Theorem: Commutative - $A \cup B = B \cup A, A \cap B = B \cap A$
- Theorem: Associative - $A \cap (B \cap C) = (A \cap B) \cap C, A \cup (B \cup C) = (A \cup B) \cup C$
- Theorem: DeMorgan - $(A \cup B)^c = A^c \cap B^c, (A \cap B)^c = A^c \cup B^c$.
- Theorem: Distributivity- $A \cap (B \cup C) = (A \cap B) \cup (B \cap C), A \cup (B \cap C) = (A \cup B) \cap (B \cup C)$
- Theorem: Absorption - $A \cap (A \cup B) = A \cup (A \cap B)$
- Theorem: Identity - $A \cap \emptyset = \emptyset, A \cup \emptyset = A$
- Theorem: $A \cup \mathcal{U} = \mathcal{U}$, $A \cap \mathcal{U} = \mathcal{A}$
- Theorem: $A \cup A^c = \mathcal{U}$, $A \cap A^c = \emptyset$
- Theorem: $(A^c)^c = A$
- Theorem: If $|A| \in \mathbb{N}$, then $|\wp(A)| = 2^{|A|}$

Paradoxes

The use of the unrestricted principle of comprehension/abstraction leads to several paradoxes in informal set theory the most famous of which is Russell's paradox.

Russell (1901) and Zermelo: Let $A = \{x | x \notin x\}$. Is $A \in A$? Both the assumption that A is a member of A and A is not a member of A lead to a contradiction (If $R = \{x | x \notin x\}$ then $R \in R$ iff $R \notin R$). Three popular forms of this paradox are:

- Does the set of all sets contain itself?
- Is there is a bibliography that lists all bibliographies that don't list themselves.
- In a village, there is a barber (a man) who shaves all those men who do not shave themselves. Who shaves the barber?

See Chapter **??** for more information about paradoxes.

Exercises

1. If operations for bags similar to those for sets are defined, what would be some of the differences?
2. What can be said about the sizes of the resulting sets?
3. What are the properties of the set operations?

2.2 Relations

Let $D_0, ..., D_n$ be sets

- *Cartesian Product*: $D_0 \times ... \times D_{n-1} = \{(a_0, ..., a_{n-1}) | a_i \in D_i; i = 0, ..., n-1\}$
- *Relation*: $R \subseteq D_0 \times ... \times D_{n-1}$ R is called an n-ary relation. Its elements are called n-tuples.

Special properties of binary relations $R \subseteq DD$. An element of R, (a, b) is called an *ordered pair*.

Reflexive R(x,x) for every x, y in D. Examples: $=, \leq$

Irreflexive not R(x,x) some x in D. Example: $<$

Symmetric R(x,y) implies R(y,x) for every x, y in D.

Antisymmetric R(x, y) and R(y, x) implies x = y for every x, y in D.

Transitive If R(x, y) and R(y, z) implies R(x, z) for every x, y, and z in D.

Equivalence R is reflexive, symmetric, and transitive. An equivalence relation partitions a set into a collection of sets called an *equivalence classes*.

Closure The closure of R with respect to property P is the smallest relation that has property P and contains R.

Order relations

A relation R is an order relation on a set D is

Quasi order transitive and reflexive

Partial order *(poset)* transitive, antisymmetric, reflexive

Linear order (or simple, or total) partial order and comparable i.e., for every x, y in D, either R(x, y) or R(y, x). This property is the *trichotomy law*: for any $a, b \in S$, either $a \leq b$ or $b \leq a$.

Well order linear order and every nonempty subset of D has a least element (l is a least element of S iff for every x in S, R(l, x)).

Other important definitions

(maximal, minimum) element

(upper, lower) bound

(least upper, greatest lower) bound

equivalence class: $\{x|R(x,a)\}$ **for R an equivalence relation**

2.3 Functions

Let X and Y be non-empty sets. A *function* $y = f(x)$ is defined on X if the rule *f* assigns to each element x of X a specific element y of Y. The set X is called the *domain* of *f* and the set Y is called the *range* (or *codomain*) of *f* and y is called the *image* of x under the rule *f* or the *value* of *f* at the element x. The rule *f* is often called a *mapping* or *transformation* or *operator*. If Y is a singleton set, then *f* is called a *constant function*.

The set of all functions from X to Y is denoted by $X \rightarrow Y$ and $f : X \rightarrow Y$ denotes *f* as an element of that set.

Let $W \subseteq X$ and $Y \subseteq Z$ and $g : W \rightarrow Y$ and $f : X \rightarrow Z$. Then f is called an *extension* of g and g a *restriction* of f.

Let $f : X \rightarrow Y$, $Y \subseteq Z$

If $A \subseteq X$, $f : X \rightarrow Y$, then the *image* of $f(A)$ is the subset of Y defined by $f(A) = \{f(x)|x \in A\}$

The function f is a mapping of X *into* Y if $f(X) \subseteq Y$.

The function f is a mapping of X *onto* Y if $f(X) = Y$; if for each y in Y there is an x in X such that f(x) = y. Such functions are also called *surjections*.

The function f is a *one-to-one* mapping of X into Y if for every x and y in X, $f(x) = f(y)$ implies $x = y$. Alternately, for every x and y in X, if $x \neq y$ then $f(x) \neq f(y)$. Such functions are also called *injections*.

If $f : X \rightarrow Y$ is both one-to-one and onto, then the *inverse mapping* $f^{-1} : Y \rightarrow X$ is defined as follows if $y = f(x)$ then $f^{-1}(y) = x$.

A function which is both one-to-one and onto is called a *one-to-one correspondence*.

Let $f : Y \to Z$ and $g : X \to Y$ then the *composition* of f and g, $f \circ g : X \to Z$, is defined by $f \circ g(x) = f(g(x))$.

A function f from A to B is called a *bijection* if it is one to one and onto, i.e. bijections are functions that are injective and surjective.

A function f from A to B is called a *partial function* if for some x in A, f(x) is undefined (i. e., f(x) is not in B).

A function f from A to B is called a *total function* if for all x in A, f(x) is defined (i. e., f(x) is in B).

Morphisms

Morphism A morphism is a map between two objects in an abstract category.

Homomorphism A general morphism is called a homomorphism. A operation preserving mapping between two objects of the same type.

Isomorphism A bijective morphism is called an isomorphism. It preserves sets and relations. An isomorphism between an object and itself is an automorphism.

Homeomorphism A continuous function between two structures that has a continuous inverse.

2.4 References

Halmos, Paul R. Naive Set Theory

Smullyan, R. M. and Fitting, Melvin Set Theory and the Continuum Problem

Part I

Logic: Syntax and Semantics

Chapter 3

Formal Systems and Logic

> Logic is a science of truths - Quine 1950
> Logic is a science of deduction - Hacking 1979

3.1 Introduction

A *formal system* consists of a *relational structure*, a formal *language*, and a *theory*. The theory describes and is used for reasoning about the properties of the relational structure. Literary interpretation, doctrines, evolution, program specifications are examples of theories with literary works, sacred scriptures, the natural world, and computer programs as the respective relational structures. The formal language (as it is described here) is the language of formal logic extended with axioms which describe the relational structure. Logic itself is formalizable as a formal system and is used to illustrate the concepts (see Figure 3.1). Figure 3.2 provides a simple view of formal systems and examples from the humanities, mathematics, and computer science.

The study of formal systems naturally separates into two areas, *proof theory* which is concerned with language, axioms, inference, and what is provable and *model theory* which is concerned with the relationship between the structure and the language and whether what is true is provable. Since the language and its axioms often become the subject of study rather than the structure, the relational structure is called an interpretation or model of the axioms. This is a reverse of the terminology used in scientific theories where the theory is said to model reality.

The clean separation of language and relational structure is an important aspect of formal systems. A set of *axioms*, which are expressions in the language and are usually called a *theory*. The axioms are a *syntactic* representation of the relational structure and its properties of interest. A theory may describe more than one relational structure. In which case, the theorems proved about

Symbols	The alphabet of a language
Formulas	Formed from strings of symbols according to the following rules: f is an atomic formula $\mathbf{F} ::= f \mid \neg F \mid F \wedge F \mid F \vee F \mid F \rightarrow F \mid F \leftrightarrow F$ – *propositional formulas* $\mid \Box F \mid \Diamond F \mid ...$ – *modal formulas* $\mid \forall x.F \mid \exists x.F$ – *first-order formulas*
Relational Structure	A set of objects and relationships among the objects.
Interpretation	A *interpretation* I is a mapping between the formulas and a relational structure i.e., I : Formulas→Structure.
Axioms	A set of formulas which describe the relational structure.
Inference rules	Inference rules describe the way formulas are derived from other formulas. They have the form: $\dfrac{\text{From: } H_0, ..., H_n}{\text{Infer: } C}$ where the H_i are called *hypotheses* or *premises* and C the *conclusion*. If the hypotheses are satisfied, then the conclusion is satisfied as well.

Figure 3.1: Concepts at a glance

Language	**Semantic Mapping**	**Relational Structure**
L	$\rightarrow$	S
Pure logic		false, true
Literary interpretation		Literary work
Algebraic theory of groups		A group
Regular expressions		Finite automata
Context-free language		Push-down automata
Context-sensitive language		Linear bounded automata
Unrestricted language		Turing machines
Program specification		Computer program

Figure 3.2: Examples of Formal Systems

one structure are true in the others. For example, group theory describes the properties of a wide variety of relational structures including some of the properties of the integers and some properties of matrices. The axioms of Geometry without the parallel axiom may be used to reason about both Euclidean and non-Euclidean universes. In the social sciences and the humanities it is common to have more than one theory for a given relational structure.

Formalized natural number arithmetic provides a simple example of a formal system.

- The relational structure is the natural numbers: 0, 1, 2, ..., with addition and other relations being the items of interest.
- The language is that of logic extended with equality and special symbols for the operations of interest. The Peano axioms for the natural numbers describe the relations of interest.
  - A syntactical representation of the natural numbers is given in terms of a successor function: $N ::= 0|s(N)$
  - The operations are defined recursively. For example, the definition of addition is given by the two equations:
    - 0 + s(n) = s(n)
    - s(m) + s(n) = s(m + s(n))
  - etc.

Aside: In the mathematical sciences, the approach is to try to create a single theory for multiple relational structures while in the humanities, multiple theories are created for a single relational structure (such as multiple interpretations for a single literary work). This often frustrates students in their general studies program. Some wonder why there is a multiplicity of interpretations of a literary work and wish that there were such a thing as the true interpretation. While others are frustrated by the rigid, symbolic methods in the mathematical sciences.

Relational structures: Relational structures, which consist of objects and relationships among the objects, may be simple or complex, static or dynamic, real, fictional, or speculative, and subjective or objective. The relational structure is also called a *interpretation,* or *model* of the axioms.

Structures can vary enormously in their internal composition and relationships. The arithmetic of the natural numbers has a simple linear structure, 0, 1, 2, 3, ... with three fundamental relations, addition, multiplication, and less than. The natural ecology of the Earth has an enormous variety of multi-layered and interacting relationships. The questions that should be asked about structures include:

1. Their size and structure.
2. Issues of homogeneity and heterogeneity.
3. Whether it is static or dynamic and if dynamic if it is deterministic or non-deterministic.
4. The precision with which the structures are known.
5. The correctness of the knowledge of the structure.
6. The completeness of the knowledge of the structure.

Language: The logic languages are objective and formalized languages (both natural and symbolic) that are used to describe relationships present in the domain of interest (a relational structure) and to communicate the formalized argumentation called proofs. They are completely symbolic language (a set of strings of symbols called formulas) built according to certain rules from an alphabet of specified primitive symbols. The symbols are taken to be merely marks which are to be manipulated according to given rules which depend only on the form of the expression composed from the symbols. The relational structure (either constructed, discovered, or in mind when the language is designed), described by the language, is called a *model* or is said to model formulas in the language.

The questions that should be asked about a language include:

1. Is it expressive enough? Can we express all of the relationships of interest?
2. It is efficient? Is there an economy of expression that facilitates ready understanding?

For example, in some situations, people prefer face to face meetings because written language is either insufficiently expressive to capture the nuances of intended meaning or inefficient in conveying that information. Film, theater, fine art, and music have their own languages.

Relationships: The relationship between the language and the structure varies in terms of the degree of precision, accuracy, objectivity, verifiability involved and ranges from accurate one-to-one decidable relationships to subjective, vague and unverifiable relationships. This range is represented in the academic disciplines of mathematics, natural science, social science, humanities and religion.

The semantic relations map expressions in relational languages to relational structures. In a sense, each human being has a mapping between language and the natural world. The questions that should be asked about semantic functions include:

1. Is the mapping between the language and structure, vague, precise, and or accurate?

2. Can the mapping be validated? Is it objective? Is it computable/decidable?

The usual use of a logic consists in translating information from some domain into the language of the logic where reasoning can be conducted without thinking about the meaning of each formula. The resulting inferences are then translated back into the domain of interest.

Axioms: The non-logical axioms describe the features and relationships of interest in the relational structure. Often the relational structure contains an infinite number of objects and to be able to facilitate the description of the relational structure, axioms are formulated as *axiom schemas* so as to limit the number of axioms to a finite number.

> **Aside:**If a simple correspondence were all that there was between a language and a relational structure, than mathematics would have none of its power and influence and would resemble biological field observations. Mathematics gets its power from the fact that its relational structure is static and infinite while many interesting properties are finitely describable. Furthermore, many of its structures are easy to map to real world domains e. g., financial accounting. In contrast, works of literature often require a literary interpretation to supply the missing and speculative relationships required for full appreciation of the text.

The questions that should be asked about the non-logical axioms include:

1. Are the number of axioms as few as possible? Is there are finite axiomatization of the relational structure?
2. Are the axioms independent of each other?
3. Is the set of axioms complete?
4. Does the set of axioms facilitate proofs of important properties of the relational structure.

Deduction: Reasoning (deduction or inference) in the language is a syntactic operation in which sentences are constructed from other sentences by purely syntactic pattern recognition. For example, the *modus ponens* pattern deduces B from the formulas A and $A \rightarrow B$.

On the language side, logic is based on deduction (or formal derivability), a method of exact inference with the advantage that its conclusions are exact − there is no possibility of mistake if the rules are followed exactly. The accuracy of deduction depends on information that is complete, precise, and consistent. The questions that should be asked about deduction includes:

1. Are the deductions sound i.e., are theorems true?

$$\frac{\text{From: } A, A \rightarrow B}{\text{Infer: } B}$$

Figure 3.3: Modus Ponens

2. Are all valid (true) formulas provable?

A formal system consists of

- a relational structure
- a language L
- a set of axioms Ax which describe the structure (*non-logical axioms*) and a
- logic that consists of a
  - finite set of *inference rules* and
  - *logical axioms*.

The logic is domain independent information while the non-logical axioms are domain specific information.

- The primary purpose of a formal system is to provide a framework for proving theorems.
- The important problem for any formal system F is find a necessary and sufficient condition that a formula of F be a theorem of F.
- When a formal theory is the object of study, it is called the **object language**. A language used to describe and analyze an object language is called a **metalanguage**. A theorem *about* a formal theory is called a **metatheorem** to distinguish it from a theorem *of* the theory.

Figure 3.4 is and elaboration of Figure 3.1 which summarizes these elements of a formal system.

Issues in Model Theory and Proof Theory

The study of formal systems naturally separates into two areas, *proof theory* which is concerned with language, axioms, inference, and what is provable and *model theory* which is concerned with the relationship between the structure and the language and whether what is true is provable. Perhaps the most significant result in this area is Gödel's incompleteness theorem the implications of which are captured in this quote:

Syntax	**Semantics**	
	Semantic Mapping	*Relational Structure*
Formulas e.g., $F ::= \mathbf{f} \mid P \mid \to FF \mid \forall x.F$ **Theorems** non-Logical Axioms of Theory T Tautologies Logical Axioms	$\rightarrow$	A set of constants and a set of relations defined on the set of constants. 1. A nonempty set A called the universe. 2. A set of n-ary functions from A^n to A, n=0, 1, ... 3. A set of n-ary relations in A^n, n=0, 1, 2, ... In pure logic the relational structure is the set $\{0, 1\}$ where 0 is interpreted as false and 1 as true. The semantic mapping is called a *valuation function* which is a total function from the set of atomic formulas to the set $\{0, 1\}$
Inference rules e.g., modus ponens **Modus ponens** $\dfrac{\text{From: } A, A \rightarrow B}{\text{Infer: } B}$		

Figure 3.4: Formal System

$\mathcal{S} = \langle \mathcal{C}, \mathcal{R} \rangle$ is a simple relational structure where

$\mathcal{C} = \{c_0, c_1, ...\}$ – A possibly infinite set of constants.
$\mathcal{R} = \{R_j^i \mid R_j^i \subset \mathcal{C}^i\}$ – A possibly infinite set of relations

Figure 3.5: Relational Structure S

> A completely unfree society (i.e., one proceeding in everything by strict rules of "conformity") will, in its behavior, be either inconsistent or incomplete, i.e., unable to solve certain problems, perhaps of vital importance. Both, of course, may jeopardize its survival in a difficult situation. A similar remark would also apply to individual human beings. - Gödel, quoted in Wang, H. *A Logical Journey: from Gödel to Philosophy.* The MIT Press. 1996.

In model theory, relational structures are the objects of primary interest. Language is introduced in order to discuss the properties of the structure. Thus, model theory begins with a class of set-theoretic objects called *relational structures* and constructs a language and a mapping from the language to the structure. Figure 3.5 contains a formal description of a relational structure.

A formal definition of the *language*, L is given in Figure 3.6.

The *semantic mapping* I between the language L and the *relational structure* S is, for the most part, self explanatory. The constants in the language are mapped to the constants in the structure and the atomic formulas are mapped to elements (tuples) of the relations. The structure and mapping $M = \langle S, I \rangle$ are then *interpretations*, or *models* of the formulas of the language. A formula f, of the language L, is *valid* in the model if it satisfies f, $M \models f$. The details of this are found in Figure 3.7.

Definitions

- A formula f of L is *valid in* M, $M \models f$, under the conditions listed in Figure 3.7.
- A set of formulas T of L is *valid* in M, $M \models T$ if every formula f, in T is valid in M, $M \models f$. A set of sentences Ss, is **satisfiable** if there is a structure A in which all of the members of Ss are true. Such a structure is called a **model** of Ss. If Ss has no model, it is **unsatisfiable**.
- A formula f of L is a **tautology**, $\models f$, if it is valid in all models.
- A formula S of L is a **logical consequence** of a set of formulas T of L ($T \models S$), if S is valid in every model of T.

The set of atomic formulas, $\mathcal{P}$, is defined by

$\mathcal{P} = \{P^i_j t_k ... t_{k+i-1} \mid t_l \in \mathcal{C},\ i,j,k,l \in \mathbb{N}\}$ with $\mathbf{f} \in \mathcal{P}$ where
$\mathcal{C} = \{F^i_j t_k ... t_{k+i-1} \mid t_k \in \mathcal{C},\ i,j,k \in \mathbb{N}\}$ is a set of *terms*,
$\{P^0_j \mid j \in \mathbb{N}\}$ is a set of *propositional constants*, and
$\{F^0_j \mid j \in \mathbb{N}\}$ is a set of *individual constants*.

The set of formulas, $\mathcal{F}$, is defined by

$\mathcal{F} ::= \mathcal{P} \mid\ \rightarrow \mathcal{F}\mathcal{F} \mid \Box\mathcal{F} \mid \forall x.[\mathcal{F}]^x_t$

where $\mathcal{V} = \{x_i \mid i \in \mathbb{N}\}$ is a set of *individual variables*, $t \in \mathcal{C}$, $x \in \mathcal{V}$, and textual substitution, $[F]^t_x$, is a part of the meta language and designates the formula that results from replacing each occurrence of t with x.

Additional operators and infix notation:

$(A \rightarrow B) \equiv\ \rightarrow AB$
$\neg A \equiv (A \rightarrow \mathbf{f})$
$(A \vee B) \equiv (\neg A \rightarrow B)$
$(A \wedge B) \equiv \neg(A \rightarrow \neg B)$
$\mathbf{f} \equiv (A \wedge \neg A)$
$(A \leftrightarrow B) \equiv ((A \rightarrow B) \wedge (B \rightarrow A))$
$\Diamond A \equiv \neg\Box\neg A$
$\exists x.A \equiv \neg\forall x.\neg A$

Figure 3.6: The syntax of the formulas of language L

We say that $\mathcal{M}$ models the formula A ($\mathcal{M}$ is a model for A) and write $\mathcal{M} \models A$ if $A \neq \mathbf{f}$ and the following hold:

- if $A \in \mathcal{P}$ and $v(A)$ is satisfied.
- if A is $\neg B$ and not $M \models B$
- if A is $\rightarrow BC$ and either $M \models C$ or not $M \models B$
- if A is $\forall x.B$ and for all constants, t, $M \models [B]^t_x$

Figure 3.7: The semantics of the formulas of L

$v : \mathcal{P} \to \{0, 1\}$	v is boolean valued as in classical logic
$v : \mathcal{P} \to [0, 1]$	v is infinite valued and is suitable for use in a probabilistic, fuzzy, and other continuous logics.
$v : \mathcal{P} \to \{\perp, 0, 1\}$	v identifies undefined formulas such as the truth of $y > 1/x$.
$v : \mathcal{P} \to \{0, ..., n\}$	v is a multivalued logic with a range of values such as a Likert scale for tests.

A valuation function v satisfies the following:

$$\begin{array}{lcl} v(\mathbf{f}) & = & 0 \\ v(p) & = & \left\{ \begin{array}{ll} 0 & \text{if } p \text{ is false} \\ 1 & \text{if } p \text{ is true} \end{array} \right. \quad \text{where } p \in \mathcal{P}. \end{array}$$

v may be extended to non atomic formulas as follows:

$$\begin{array}{lcl} v(\neg A) & = & 1 - v(A) \\ v(A \vee B) & = & \max(v(A), v(B)) \\ v(A \wedge B) & = & \min(v(A), v(B)) \\ v(A \to B) & = & \max(v(\neg A), v(B)) \\ v(\exists x.A) & = & \max_{c \in C}(v([A]^c_x) \\ v(\forall x.A) & = & \min_{c \in C}(v([A]^c_x) \end{array}$$

Valuation functions and the models relation in classical logic.

$$v \models A \quad \text{iff} \quad v(A) = 1$$

Figure 3.8: Valuation functions for logic

In pure logic there is no relational structure so in classical logic there are precisely two **truth values**: **true** and **false**. A **valuation function** v maps atomic formulas to truth values (see Figure 3.8). Fuzzy and probabilistic logics are examples of continuous valued logics where the truth values are in the closed interval [0, 1].

The issues of interest are:

- *Consistency 1*: Given a set of formulas T, does it have a model i. e. exist an M such that $M \models T$?
- *Soundness*: Do the proof rules lead to valid theorems i. e., does $M \vdash F$ imply $M \models F$?
- Given a set of formulas, how large a structure is required to satisfy the set of formulas?

- *Decidability*: Given a theory T, is the theory decidable i. e., for every formula F in T, is either F or $\neg F$ a theorem? Alternatively, for formula F, is either $\vdash F$ or $\vdash \neg F$?

- *Consistency 2*: Given a theory T and a formula F are both F and $\neg F$ provable? If so the theory is said to be *inconsistent*.

- *Completeness*: Are the set of formulas and rules of inference sufficient to insure that every valid formula is a theorem i. e., does $M \models F$ imply $M \vdash F$?

We begin with semantics and model theory and follow with syntax and proof theory.

3.2 Semantics and Model theory

Multiple World Semantics (Saul Kripke)

A formula F is **valid** (a **tautology**), $\models F$, iff for all **w** in **W**, $M \models F$ i.e., F *is true in all possible worlds.* A formula F is said to be **valid** ($\models F$) iff it is valid in all models M ($M \models F$ for all M). A valid formula is called a **tautology**.

Predicate Logic (or Predicate Calculus or First-Order Logic) is a generalization of Propositional Logic.

Generalization requires the introduction of variables.

Chapter 7 contains more details about relational structures.

As stated earlier, semantic relations map formulas to objects in a semantic structure. The mapping may be 1-to-1 giving an accurate representation. However, it is not always possible to construct accurate representations. For example, the word “red” is mapped to a number of actual colors and ambiguously to many others. Most mappings in scientific theories are vague. Bertrand Russell’s paper on vagueness (in Chapter 30) is a highly readable discussion of these ideas. For a taxonomy of semantic relations see Section 27.2.

Additional information on multivalued logics is ¡a href=”MultiValued.html”¿available¡/a¿.

In pure logic, the semantics are often given by *valuation functions* which are total functions from the set of atomic formulas to the set 0, 1 where 0 is interpreted as false and 1 as true. The valuation function may be extended to all formulas in Figure 3.6. A formula is said to be **satisfiable** if there is a validation function which make it true. A formula is said to be a **contradiction** if there is a validation function which makes it false. A formula A is said to be **valid** (a **tautology**), —=A, if it is true for all valuations v. For propositional formulas, ¡a href=”truthTables.html”¿truth tables¡/a¿ are an accepted method to determine whether a formula is a tautology (valid), satisfiable, or a contradiction. Boolean semantics are based on the coherence theory of truth.

¡a href=”../Philosophy/Epistemology/Correspondence.html”¿More details about correspondence relations.¡/a¿

In model theory the concern is with the following terms and concepts:

Satisfiable A formula F that is true for some structure M, $M \models F$, is said to be *satisfiable*.

Valid A formula F that is true in all structures is said to be *valid*, $\models F$ (i.e. if $M \models F$ for all M, then F is valid). The notation $T \models f$ is extended to mean that the formula f is true in all those structures in which the axioms of T are true.

Tautology A formula that is valid is called a *tautology.*The formula *A or not A* is a tautology in classical logic.

Contradiction A formula that is not satisfiable in any structure is called a *contradiction* (i.e., if not $M \models F$ for all M, then F is a contradiction). The formula *A and not A* is a contradiction in classical logic.

Sound The inference rules in a language are said to be *sound* iff every theorem derived by an inference rule is valid (i.e. for each formula F, if $\vdash F$ then $\models F$).

Complete A language is *complete* iff every valid formula is a theorem (i.e., for each formula F, if $\models F$ then $\vdash F$).

Consistent A theory (a language) is *consistent* if it has a model.

The sentence X of language L **follows logically** from the sentences of the class K if and only if every model of the class $\mathbf{K}$ is a model of the sentence X (Tarski).

Chapter 6 contains more details about semantics and model theory.

3.3 Syntax and Proof theory

David Hilbert initiated the study of **metamathematics** or **proof theory** which is the study of formal theories using the following methods:

- metamathematics should belong to informal and intuitive mathematics
- the metatheorems must be understood and the deductions must carry conviction
  - the axiom of choice must not be used
  - the methods used must be finitary
    - this excludes the consideration of infinite sets as completed entities and

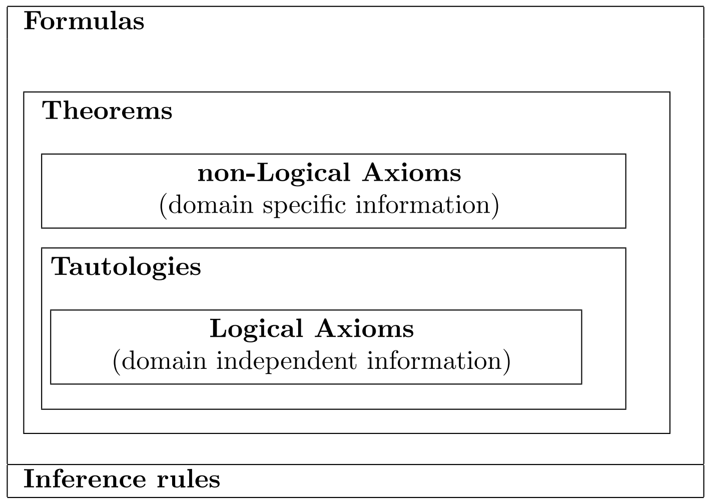

Figure 3.9: The domain of proof theory

    * requires that existence proofs provide an effective procedure for constructing the object.
    * mathematical induction is admissible as a finitary method of proof as it does not require the notion of completed infinity of the natural numbers.
  - Only a minimal fragment of natural language is taken as the meta-language.

Hilbert's approach requires an *effective procedure (decision procedure)* which is a set of instructions that provides a mechanical means by which the answer to any one of a class of questions can be obtained in a finite number of steps. Therefore,

- Formulas are *finite* strings of symbols so that the notion of a *formula* is effective. We can decide whether a finite sequence of symbols is a formula.
- The notion of *axiom* must be effective so that we can decide whether an arbitrary formula is an instance of an axiom. The aesthetic values for sets of axioms include:
  - The axioms are independent of each other (a reductionist approach).
  - The set of axioms is minimal.
- The notion of *inference* must be effective so that we can decide whether each member of a sequence of formulas may be inferred from one or more

of those preceding it by a rule of inference. This implies, also, that the notion of *proof* is effective – we can decide whether or not a finite sequence of formulas is a proof. The aesthetic values for inference rules include:

  - *Soundness*: Inference rules must be sound (truth preserving).

  - Inference rules must be complete in the sense of dealing with each of the logical connectives.

- The notion of *theorem* is not required to be effective otherwise the theory would lose its appeal to mathematicians. For example, boolean algebra is of little interest to mathematicians since it is a decidable theory however, it is of considerable interest to engineers who use it to design digital systems.

The symbols are taken to be merely marks which are to be manipulated according to given rules which depend only on the form of the expression composed from the symbols.

- For mathematics, the principle reason for formulating formal axiomatic theories is that fundamental questions of consistency and completeness can be discussed in a precise and definitive way because the notion of proof is made explicit.

- For computer science, the principle reason is to facilitate computation by computers.

Let L be the language of first-order logic, T be a set of formulas of L, A_T, F_T, and P_T be, respectively, the sets of constants, function symbols, and predicate symbols appearing in the formulas of T. Let Tr be the set of terms of the language. A *ground term* (respectively, *atom*, *literal*, or *sentence*) is a term (respectively, atom, literal, or sentence) which has no variables.
It is important to distinguish between logic and an application of logic.
A formal system defined syntactically in this manner has no notion of meaning or semantics. It is can be viewed as a meaningless game with symbols.
A formal system F is a theory T which is

- a collection of abstract symbols, Σ, together with

- a set of rules G (a grammar) expressing how strings of these symbols (formulas) can be combined in order to create new symbol strings (formulas),

- a set of symbol strings (formulas), called axioms, that are taken to be theorems (i.e., they are assumed without proof), and

- a set of rules of logical inference for generating new grammatically correct strings (theorems) from earlier ones

- a relational structure S and

- a semantic mapping I from the formulas to the structure S.

More details about the syntax of logic languages is found in Chapter 4.

In proof theory the concern is with the following terms and concepts:

Axiom An axiom is a formula (ultimately a statement that is true for some structure). The selection of axioms should be those that facilitate proofs.

Inference rule, sound A *rule of inference*, d, is a partial function from the power set of formulas to the set of formulas i. e., $d : \wp(F) \to F$. Example: *Modus ponens* infers B from the set $\{A, A \to B\}$. A rule of inference is said to be *sound* if whenever $d(A) = f$ and $M \models A$, then $M \models f$ i.e, it infers valid formulas.

Proof, Theorem A *proof* is a sequence of formulas each of which is an *axiom*, or may be inferred by an inference rule, from formulas appearing earlier in the sequence. A sequence of formulas $\sigma = f_0, f_1, ..., f_n$ is said to be a *deduction* (or *proof*) from a theory T of the last formula f_n (f_n is said to be a *theorem*), if each formula in the sequence is either an axiom (a member of A) or is derived from formulas preceeding it in the sequence by a rule of inference. We write $T \vdash f$ to say that f is a *theorem* and has a proof in the theory T.

Decision method A *decision method* is an effective method for deciding whether or not a formula is a theorem.

Completeness A theory T is said to be *complete*, if for every formula $f \in F$, either $T \vdash f$ or $T \vdash \neg f$ i.e., every formula has a proof or a disproof. A theory T is said to be *incomplete*, if for some formula f, $T \models f$ but it is not the case that $T \vdash f$ i.e., some formula is true but is not a theorem.

Consistent and Inconsistent A theory T is said to be *consistent* if one of the following holds (they are equivalent)

- it has a model (Gödel-Henkin completeness theorem; ¡a href=”../Philosophy/Epistemology/Correspondence.html”¿Correspondence theory of truth¡/a¿).
- not all formulas are theorems.
- for any formula f, at most one of f or $\neg f$ are theorems (Freedom from contradiction; ¡a href=”../Philosophy/Epistemology/Coherence.html”¿Col theory of truth¡/a¿).

Let Σ be a set of symbols, G be a grammar for formulas F, Let A_F be a set of *axioms* which describe the relational structure S. The set of axioms should be as small as possible and each axiom should be independent of the others. A set of axioms may describe properties common to several structures; each structure would have its own interpretation mapping(s). The set of axioms may be viewed as defining a set of ¡a href=”../Philosophy/Epistemology/IMT.html”¿invariant properties¡/a¿ of a class of structures.

Let I be a set of sound rules of inference, then $T = \langle A, I \rangle$ is a *theory*.

Completeness Theorem: Propositional and First-order logic are complete (Gödel).

Incompleteness Theorem: Elementary arithmetic is incomplete (Gödel).

Aside:*Interesting.* A theory is interesting when not all formulas of the language are theorems or the definition of a theorem is not effective. A relational structure is interesting when it is infinite, difficult to axiomatize, or difficult to construct a consistent theory.

		Source
Syntactic methods:	Axiomatic method	Aristotle, Hilbert
	Sequent method	Gehard Gentzen
Semantic methods:	Analytic tableaux	Beth, Hintikka, Kripke

Figure 3.10: Proof methods

3.4 Exercises

1. Construct a formal system for arithmetic.
2. Define an interpretation for arithmetic
3. Construct a formal system for simple algebraic equations.
4. Define an interpretation for algebraic equations
5. Show that $v(A \wedge B) = \min\{v(A), v(B)\}$.
6. Show that
   (a) all the propositional operators can be defined in terms of $\neg$, and $\wedge$,
   (b) all the propositional operators can be defined in terms of $\neg$, and $\vee$,
   (c) all the propositional operators can be defined in terms of $\neg$, and $\rightarrow$.
   (d) all the propositional operators can be defined in terms of the Sheffer stroke.

3.5 References

Fagin, Halpern, & Vardi What is an inference rule? *Journal of Symbolic Logic* **57**:3, 1992, pp. 1018-1045.

Bell, J. L. and Slomson, A. B. Models and Ultraproducts: an introduction. North-Holland Publishing Company. 1969.

Etchemendy

Robbin, Joel W. *Mathematical Logic: a first course* W. A. Benjamin, Inc. 1969.

Chapter 4

Syntax

4.1 Grammar and Language

A grammar G is a quadruple $G = \langle V, T, P, S \rangle$ where V is a finite set of symbols called variables, T is a finite set of symbols called terminal symbols disjoint from V, P is a finite set of rewriting rules (productions) of the form

$$u ::= w_0 \mid w_1 \mid ... \mid w_n$$

where u is a nonterminal and the w_i are sets or strings containing any combination of variables and terminals. S is an element of V called the start symbol.

The language of the grammar, $L(G)$, consists of all strings of symbols defined by the grammar.

4.2 The Language of Logic

The formal language of first-order logic is the set of formulas **F** in the language The set of atomic formulas, $\mathcal{P}$, is defined by

$\mathcal{P} = \{P^i_j t_k ... t_{k+i-1} \mid t_l \in \mathcal{C},\ \ i, j, k, l \in \mathbb{N}\}$ with $\mathbf{f} \in \mathcal{P}$ where
$\mathcal{C} = \{F^i_j t_k ... t_{k+i-1} \mid t_k \in \mathcal{C},\ i, j, k \in \mathbb{N}\}$ is a set of *terms*,
$\{P^0_j \mid j \in \mathbb{N}\}$ is a set of *propositional constants*, and
$\{F^0_j \mid j \in \mathbb{N}\}$ is a set of *individual constants*.

The set of formulas, $\mathcal{F}$, is defined by

$$\mathcal{F} ::= \mathcal{P} \mid\ \rightarrow \mathcal{F}\mathcal{F} \mid \Box\mathcal{F} \mid \forall x.[\mathcal{F}]^x_t$$

where $\mathcal{V} = \{x_i \mid i \in \mathbb{N}\}$ is a set of *individual variables*, $t \in \mathcal{C}$, $x \in \mathcal{V}$, and textual substitution, $[F]^t_x$, is a part of the meta language and designates the formula that results from replacing each occurrence of t with x.

Additional operators and infix notation:

$(A \to B) \equiv \to AB$
$\neg A \equiv (A \to \mathbf{f})$
$(A \vee B) \equiv (\neg A \to B)$
$(A \wedge B) \equiv \neg(A \to \neg B)$
$\mathbf{f} \equiv (A \wedge \neg A)$
$(A \leftrightarrow B) \equiv ((A \to B) \wedge (B \to A))$
$\Diamond A \equiv \neg\Box\neg A$
$\exists x.A \equiv \neg\forall x.\neg A$

The F^i_j are called *functors* or *function symbols* and i is its *arity* (the number of arguments required). The zero-ary functions, $\{F^0_j \mid j \in \mathbb{N}\}$ do not have arguments and are called *constants*. Functions are often a part of the definition of the syntax of logic however they can be replaced with additional predicates with the results of longer and more complex formulas.

The elements of $\mathcal{P}$ are called *atomic formulas*. The P^i_j are called *functors* or *predicate symbols* and i is its *arity* (the number of arguments required). The zero-ary predicates $\{P^0_j \mid j \in \mathbb{N}\}$ do not have arguments and are called *propositions*.

The prefix notation has the advantage of being unambiguous while the *infix notation* has the advantage of being more readable but requires the use of grouping symbols or precedence rules to remove ambiguity. The prefix syntax was chosen for two reasons. First to be minimal and second to avoid the use of parentheses. Informally, we use the more common infix notation with parentheses and introduce additional logical operators via syntactic definitions.

Constants: $C = \{f^0_j \mid; j \in \mathbb{N}\}$
Functions: $f^i_j(t_1, t_2, ..., t_i)$ for $f^i_j t_1 t_2 ... t_i$ and $i > 0$
Terms: the constants and functions
Predicates: $p^i_j(t_1, t_2, ..., t_i)$ for $p^i_j t_1 t_2 ... t_i$
Negation: $\neg A = \to A\mathbf{f}$
Disjunction: $(A \vee B) = \to \neg AB$
Conjunction: $(A \wedge B) = \neg \to A \neg B$
Conditional: $(A \to B) = (\neg A \vee B$
Biconditional: $(A \leftrightarrow B) = ((A \to B) \wedge (B \to A))$
Existential Quantifier: $\exists x.A = \neg\forall x \neg A$
Diamond: $\Diamond A = \neg\Box\neg A$

Formulas are pronounced as follows:

false : $\mathbf{f}$
not A : $\neg A$
A or B : $A \vee B$
A and B : $A \wedge B$
If A then B : $A \rightarrow B$
A if and only if B : $A \leftrightarrow B$
Forall x A: $\forall x.A$
Exists x such that A: $\exists x.A$
Necessarily A: $\Box A$
Possibly : $\Diamond A$

The following conventions allow a reduction in the number of parentheses:

1. Drop the outermost parentheses.
2. If other parentheses are omitted, then the operators are ranked in precedence (from high to low) as follows: $\neg, \exists x., \forall x., \wedge, \vee, \rightarrow, \leftrightarrow$.
3. To improve readability we will sometimes use brackets, [ ], and braces, { }, for grouping in addition to parentheses.

In other approaches, the functions defined by these rules are called *ground terms* and the formulas defined by these rules are called *sentences* or *closed formulas.* This requires additional definitions for what are called *free* and *bound variables.* In the approach used here, there are no formulas with free variables. All variables are introduced using the quantifiers ($\exists$ and $\forall$) and are bound variables. A *sentence* (or a *closed* formula) is a formula A such that for every variable x and every constant c, $[A]^x_c = A$. This means that in a closed formula A, all of the variables in A are quantified. In the formula $\forall x.A$, A is the *scope* of the quantifier and x is *not free* in $\forall x.A$. The substitution rules are:

Assume x, y, and t are distinct.

$[t]^x_t = x$
$[t_i]^x_t = t_i$ where $t \neq t_i$
$[P^0_i]^x_t = P^0_i$
$[F^0_i]^x_t = F^0_i$
$[P^i_j t_1 ... t_i]^x_t = P^i_j [t_1]^x_t ... [t_i]^x_t$
$[F^i_j t_1 ... t_i]^x_t = F^i_j [t_1]^x_t ... [t_i]^x_t$
...
$[A \rightarrow B]^x_t = [A]^x_t \rightarrow [B]^x_t$
...
$[\forall y A]^x_t = \forall y [A]^x_t$ where $x \neq y$
$[\forall x A]^x_t = \forall x A$
$[\exists y A]^x_t = \exists y [A]^x_t$ where $x \neq y$

	Natural language form	**Symbolic form**
A proposition	The sun is shining	P
A property	Clarence is tall.	tall(Clarence)
A binary relation	John loves Mary.	loves(John, Mary)
A complex example	There is a man that likes pizza.	$\exists x.[man(x) \wedge likes(x, pizza)]$

Figure 4.1: Examples

$[\exists xA]^x_t = \exists xA$

Alternative approaches to defining free and bound variables and there are several alternate notations for textual substitution are discussed in Chapter 24.

4.3 Inference Rules and Proof

A *theory* is a set of sentences and rules of inference (deduction). *Modus ponens* is a commonly used rule of inference.

$$\frac{\text{From: } A, A \rightarrow B}{\text{Infer: } \;B}$$

Chapter **??** provides more details on rules of inference. A theory is *monotonic* if the truth of a sentence does not change when new axioms are added to the system.

Figure 4.2 illustrates the domain of proof theory.

Definition 4.1

- *A* proof *of A from a set of formulas Ss, $Ss \vdash A$, is a sequence of statements, each of which is an element of Ss or is derived from previous statements in the sequence by means of a rule of inference. The last statement in the sequence is the goal of the proof and is called a* theorem*.*

- *A set of sentences Ss is* inconsistent *iff for some formula A there is a proof of A and a proof of $\neg A$ (i.e. $Ss \vdash A$ and $Ss \vdash \neg A$). Equivalently, a set of sentences is* inconsistent *iff it contains all sentences. A set of sentences Ss is* consistent *iff it is not inconsistent.*

- *A theory (axioms and rules of inference) is* complete *iff every formula or its negation has a proof (either $\vdash A$ or $\vdash \neg A$). Equivalently, a theory (axioms and rules of inference) is* incomplete *iff some formula and its negation does not have a proof (neither $\vdash A$ nor $\vdash \neg A$).*

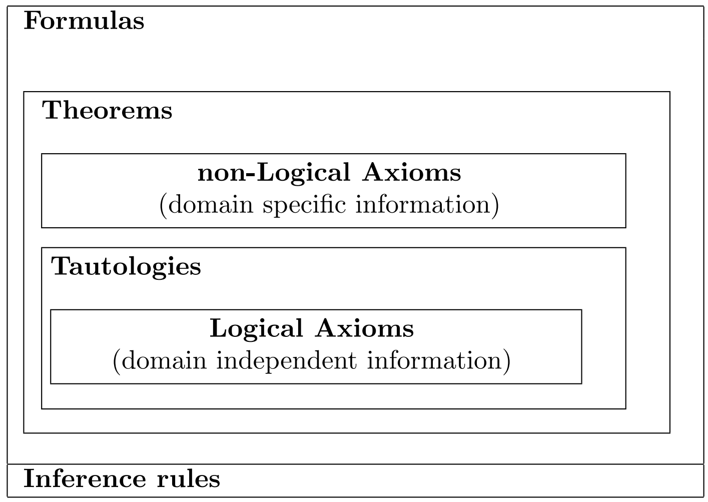

Figure 4.2: The domain of proof theory

Proofs may be constructed either bottom-up or top-down. A *bottom-up* or *forward proof* begins with the known theorems and produces new theorems. A *top-down*, *goal directed* or *backward* proof begins with the desired conclusion works back toward the known theorems.

Traditional mathematical proofs are done in a stilted form of ordinary prose which often contain gaps which the reader is expected to fill in. There are several more formal proof styles. In the *Hilbert style* (see Section 5.3) each step in the proof justified with a reason why it is true. *Natural deduction* (see Section 5.4) style proofs tend to pick apart statements and reassemble them into new statements. *Analytical style* (see Section 5.6) proofs pick statements apart until a contradiction occurs. Sequent (see Section 5.7) ... A *refutation* description? *style* proof tries to refute the statement to be proved. Refutation style proofs often construct a model of the formula as a side-effect of the proof process. This aspect is particularly useful for propositional logics which often have the finite model property. Chapter **??** provides details of these proof methods.

4.4 Propositional and First-Order Logic

Propositional Logic

Propositional Logic
The quadruple, $M = (S, \mathbf{f}, \supset, P)$, defines *propositional logic* where

- S is a set of sentences or propositions defined as follows:
  - $\mathbf{f}$ is a distinguished element of S called *falsehood*.
  - $\supset$ is a binary operation on elements of S such that if X, Y $\in$ S, then X $\supset$ Y $\in$ S.

  Note: S may contain symbols other than those composed of $\mathbf{f}$ and $\supset$. In particular, parentheses, brackets, and braces may be used for grouping and other symbols may be used as atomic propositions. Abbreviations:
  - $\neg X$ is $X \supset \mathbf{f}$
  - $X \wedge Y$ is $\neg(X \supset \neg Y)$
  - $X \vee Y$ is $\neg X \supset Y$
  - $X \equiv Y$ is $(X \supset Y) \wedge (Y \supset X)$
- The set P is defined as follows. The any formula of S of the following forms (called axiom schemas) is in P.
  1. $A \supset (B \supset A)$
  2. $[A \supset (B \supset C)] \supset [(A \supset B) \supset (A \supset C)]$
  3. $\neg\neg A \supset A$

  The set P is closed under the following rule: whenever formulas of the form A and $A \supset B$ are in P then so is the formula B. This is called an inference rule and is known as *modus ponens*.

A *valuation function* v is a function from S to $\{0, 1\}$ that satisfies the following restrictions:

1. $v(\mathbf{f}) = 0$
2. $v(A \supset B) = 0$ if $v(A) = 1$ and $v(B) = 0$ and equals 1 otherwise.

A formula A is a *tautology* if $v(A) = 1$ for every valuation function v.

If A is in P, then we write $\vdash A$. If A is a tautology, then we write $\models A$.

If Γ is a set of formulas and for every validation function v such that for each formula A in Γ, $v(A) = 1$ and $v(B) = 1$ for some formula B, then we write $\Gamma \models B$. If a set of formulas Γ is closed under modus ponens, then we write $\Gamma \vdash B$ for any formula B in Γ.

First-order Logic

Predicate Logic The septuple, $M = (S, C, \mathbf{f}, \supset, \forall, V, P, T)$, defines *predicate logic* where

- S is a set of sentences or propositions defined as follows:

- **f** is a distinguished element of S called *falsehood*.
- $P^i_j(c_{k,0}, c_{k,1}, ..., c_{k,i-1})$ are elements of S where $P^i_j \in P$, $c_{i,j} \in C$, and i, j, k are in $\mathbb{N}$. If i = 0, there are no arguments.
- $\supset$ is a binary operation on elements of S such that if X, Y $\in$ S, then X $\supset$ Y $\in$ S.
- $\forall$ is a binary operation on elements of V (a set of variables) and S such that if $A(c)$ is a formula in S with c a constant in C, x a variable in V not appearing in $A(c)$, and $A(x)$ is $A(c)$ with all occurances of c replaced by x, then $\forall x A(x) \in S$.

Note: Where necessary, we will use parentheses, brackets, and braces to indicate grouping. Abbreviations:

- $\neg X$ is $X \supset \mathbf{f}$
- $X \wedge Y$ is $\neg(X \supset \neg Y)$
- $X \vee Y$ is $\neg X \supset Y$
- $X \equiv Y$ is $(X \supset Y) \wedge (Y \supset X)$
- $\exists x A$ is $\neg \forall x \neg A$

• The set T is defined as follows. Any formula of S of the following forms (called axiom schemas) is in T.

1. $A \supset (B \supset A)$
2. $[A \supset (B \supset C)] \supset [(A \supset B) \supset (A \supset C)]$
3. $\neg\neg A \supset A$
4. $\forall x A \supset A(a)$ where $A(a)$ is A in which every occurance of x is replaced by a where a is any constant or a variable not occuring in A.

The set T is closed under the following *inference* rules:

1. *Modus ponens:* Whenever formulas of the form A and $A \supset B$ are in P then so is the formula B.
2. *Generalization:* For each formula A in T, $\forall x A$ where x is a variable is in T.

A set of formulas Γ is *inconsistent* iff $\Gamma \vdash \mathbf{f}$.

The set Γ is *consistent* iff it is not inconsistent.

NEED DEFINITIONS OF $\vdash$ and $\models$

4.5 Exercises

1. Use truth tables to validate the following equivalences:

   (a) $A \to B = \neg A \vee B$
   (b) $A \leftrightarrow B = (A \to B) \wedge (B \to A) = (A \wedge B) \vee (\neg A \wedge \neg B)$
   (c) $A \vee B = \neg A \to B$
   (d) $A \leftrightarrow B = (A \to B) \wedge (B \to A)$

2. Justify the following abbreviations
   - $\neg F$ for $F \to f$
   - $A \to B$ for $\neg A \vee B$ - called conditional or implication.
   - $A \leftrightarrow B$ for $(A \to B) \wedge (B \to A)$ or $(A \wedge B) \vee (\neg A \wedge \neg B)$ - biconditional or equivalence.
   - $\exists x.F = \neg \forall x. \neg F$
   - $\forall x.F = \neg \exists x. \neg F$
   - $\Diamond F = \neg \Box \neg F$
   - $\Box F = \neg \Diamond \neg F$

3. Often, a minimal set of operators is chosen and the other operators are introduced as abbreviations.
   - **f**, $\to$, $\forall$ or $\exists$, $\Box$ or $\Diamond$;
   - $\neg, \to$, $\forall$ or $\exists$, $\Box$ or $\Diamond$
   - $\neg, \wedge$, $\forall$ or $\exists$, $\Box$ or $\Diamond$
   - $\neg, \vee$, $\forall$ or $\exists$, $\Box$ or $\Diamond$

4. Define formulas in prefix form and give the rules for substitution.

5. For Propositional Logic, prove the following:
   (a) $\vdash A \supset A$ Hint: Read A for A, $A \supset A$ for B, and A for C in axiom b.
   (b) $\vdash f \supset A$ Hint: Use axioms a and c and the deduction theorem (if $A \vdash B$ then $\vdash A \supset B$).
   (c) $\vdash A \supset [B \supset (A \supset B)]$
   (d) $\vdash A \supset \{\neg B \supset [\neg(A \supset B)]\}$
   (e) $\vdash \neg A \supset [B \supset (A \supset B)]$
   (f) $\vdash \neg A \supset [\neg B \supset (A \supset B)]$
   (g) missing
   (h) $\vdash (A \supset B) \supset [(A \supset B) \supset B]$
   (i) missing 8 ... 23
   (j) **Soundness:** If $\vdash A$, then A is a tautology.
   (k) **Completeness:** If A is a tautology, then $\vdash B$.

6. For First-Order Logic, prove the following:

(a) $\vdash \forall x A(x) \supset \forall y A(y)$ where where y does not occur in $A(x)$ and replaces all occurances of x.

(b) **The Soundness Theorem.** If $M \vdash \Gamma$ and $\Gamma \vdash A$, then $M \models A$.

(c) **The Completeness Theorem:** The set Γ is consistent if and only if Γ has a model.

(d) **Corollary (Gödel 1931)** A formula A is valid if and only if $\vdash A$.

add exercises for formula simplification

Chapter 5

Proof Methods in Logic

5.1 Preliminaries

Let Σ be a set of symbols and Σ^* be the set of all strings of finite length composed of symbols in Σ including the empty string. A language $\mathcal{L}$ is a subset of Σ^*. Alternately, let $G = \langle \Sigma, P, S \rangle$ be a grammar where Σ is a set of symbols, P a set of grammar rules, and S the symbol for sentences in the language. The notation $\mathcal{L}(G)$ designates the language defined by the grammar G. The set of strings in $\mathcal{L}/\mathcal{L}(G)$ are called *sentences* or *formulas*.

Three sets of formulas are distinguished, *axioms* ($\mathcal{A}$), *theorems* ($\mathcal{T}$), and *formulas* ($\mathcal{F}$). In monotonic logic systems the relationship among them is:

$$\mathcal{A} \subset \mathcal{T} \subset \mathcal{F} = \mathcal{L} \subset \Sigma^*$$

If the set of theorems is the same as the set of formulas ($\mathcal{T} = \mathcal{F}$), then the system is of little interest and in logic is said to be contradictory. Inference rules $\mathcal{I}$ are functions from sets of formulas to formulas ($I : \wp(\mathcal{L}) \to \mathcal{L}$ for each $I \in \mathcal{I}$). The set of theorems are constructed from the set of axioms by the application of rules of inference. A *proof* is a sequence of statements, each of which is an axiom, a previously proved theorem, or is derived from previous statements in the sequence by means of a rule of inference. The notation $U \vdash T$ is used to indicate that there is a proof of T from the set of formulas U. The task of determining whether or not some arbitrary formula A is a member of the set of theorems is called *theorem proving*.

There are several styles of proofs. The semi-formal style of proof common in mathematics papers and texts is a paragraph style. Formal proofs are presented in several formats. The following are the most common.

- Hilbert style proofs
- Natural Deduction

- Analytic Tableaux
- Sequent Systems

Axiom systems have many logical axioms and few inference rules. Natural deduction systems have no logical axioms, only inference rules. Sequent and Tableaux systems have one logical axiom and many inference rules. I begin with the axiomatic approach since it is the most familiar.

- *Direct proofs.* In a direct proof, the last statement in the sequence is the goal of the proof. A direct proof of a statement A begins with what is known, various assumptions, axioms, and previously proved theorems. At each step, the consequences of what is already known are explored. The proof terminates when the statement A is derived through the application of a rule of inference. For a formula of the form $A \rightarrow B$, proof begins with the assumptions encapsulated in A, and proceeds to construct a sequences of statements each of which is an axiom, a previously proved theorem, or follows from previous statements by a rule of inference. The last statement in the sequence is B. However, it is easy to become diverted from the path to the goal B. Direct proofs are also called *bottom-up* proofs.

- *Indirect proofs:* In an indirect proof, the first statement is the negation of the statement to be proved. An indirect proof of the statement A begins with the assumption that the statement is false, i.e., assume that $\neg A$ is true. The goal is to show that this assumption leads to a contradiction. At each step, the question is asked, "What do I need to know in order for the goal to be true?". The answer supplies intermediate goals. The proof terminates when all goals end in a contradiction. Indirect proof is also known as *proof by contradiction*, *top-down proof*, *goal directed proof*, and *backward-chaining*.

- *Prove an equivalent expression:* To prove A given that $A \equiv B$, prove B instead. A commonly used equivalence is to prove the contrapositive i.e., to prove $A \rightarrow B$, prove $\neg B \rightarrow \neg A$ instead.

- *Proof by counterexample*: Given an assertion of the form : $\forall x P(x)$, disprove it by showing that there is a c such that $\neg P(c)$. This is equivalent to a direct proof of $\exists x \neg P(x)$.

- *Mathematical induction.* Mathematical induction is an axiom schema of the form: if $P(0) \wedge [P(n) \rightarrow P(n+1)]$ then $\forall n P(n)$. To use it, show that $P(0)$ holds and then assume $P(n)$ and show $P(n+1)$.

  In second order logic, it is a single axiom and it has the form: $\forall P\{$ if $P(0) \wedge [P(n) \rightarrow P(n+1)]$ then $\forall n P(n)\}$ In a proof using induction, the establishment of P(0) is called the base step. The assumption of P(n) is called the inductive hypothesis, and the proof of P(n+1) is called the induction step.

Strong induction combines the base step and the inductive hypothesis in the assumption that $P(i)$ holds for all $i < k$ and then the inductive step requires proof that $P(k)$ holds.

- *Recursive mathematical definitions.* Recursive definition are of the form:
    1. List the basic elements of the set.
    2. Provide rules for defining additional elements of the set. The rules utilize the basic elements and the rules.
    3. There are no elements other than those constructed under rules 1 and 2.
- *Well orderings.* Every nonempty set of a linear order contains a smallest member.
- The formulas of logic are defined in Figure 5.1. Terminology: implication, converse, inverse, contrapositive, negation, and contradiction
    - Proof or Theorem: $A \vdash B$ proof of B from A.
    - Implication: $A \rightarrow B \equiv \neg A \vee B$
        * Converse: $B \rightarrow A$
        * Inverse: $\neg A \rightarrow \neg B \equiv B \rightarrow A$
        * Contrapositive: $\neg B \rightarrow \neg A \equiv A \rightarrow B$
    - Negation: $\neg\neg A \equiv A$; $\neg(A \vee B) \equiv \neg A \wedge \neg B$; $\neg(A \wedge B) \equiv \neg A \vee \neg B$; $\neg(A \rightarrow B) \equiv A \wedge \neg B$; $\neg(A \leftrightarrow B) \equiv (A \wedge \neg B) \vee (\neg A \wedge B)$; $\neg \forall x A \equiv \exists x \neg A$; $\neg \exists x A \equiv \forall x \neg A$.
    - Contradiction: $A \wedge \neg A$

5.2 The Axiomatic Method

Axiom systems have few inference rules and often many axioms and reason *forward* (or *bottom up*) from axioms to theorems by applications of the inference rules. The disadvantage with forward reasoning is that it gives no insight on how to prove an arbitrary formula, thus requiring (considerable) experience. Proofs, however, are often shorter than those in other reasoning systems.

Substitution
Modus Ponens

5.2.1 Classical logic

Axioms

1. $\rightarrow A \rightarrow BA$

The set of atomic formulas, $\mathcal{P}$, is defined by

> $\mathcal{P} = \{P^i_j t_k ... t_{k+i-1} \mid t_l \in \mathcal{C},\ \ i,j,k,l \in \mathbb{N}\}$ with $\mathbf{f} \in \mathcal{P}$ where
> $\mathcal{C} = \{F^i_j t_k ... t_{k+i-1} \mid t_k \in \mathcal{C},\ i,j,k \in \mathbb{N}\}$ is a set of *terms*,
> $\{P^0_j \mid j \in \mathbb{N}\}$ is a set of *propositional constants*, and
> $\{F^0_j \mid j \in \mathbb{N}\}$ is a set of *individual constants*.

The set of formulas, $\mathcal{F}$, is defined by

> $\mathcal{F} ::= \mathcal{P} \mid\ \rightarrow \mathcal{F}\mathcal{F} \mid \Box\mathcal{F} \mid \forall x.[\mathcal{F}]^x_t$
>
> where $\mathcal{V} = \{x_i \mid i \in \mathbb{N}\}$ is a set of *individual variables*, $t \in \mathcal{C}$, $x \in \mathcal{V}$, and textual substitution, $[F]^t_x$, is a part of the meta language and designates the formula that results from replacing each occurrence of t with x.

Additional operators and infix notation:

> $(A \rightarrow B) \equiv\ \rightarrow AB$
> $\neg A \equiv (A \rightarrow \mathbf{f})$
> $(A \vee B) \equiv (\neg A \rightarrow B)$
> $(A \wedge B) \equiv \neg(A \rightarrow \neg B)$
> $\mathbf{f} \equiv (A \wedge \neg A)$
> $(A \leftrightarrow B) \equiv ((A \rightarrow B) \wedge (B \rightarrow A))$
> $\Diamond A \equiv \neg\Box\neg A$
> $\exists x.A \equiv \neg\forall x.\neg A$

Figure 5.1: Formulas of Logic

2. →→A→BC→→AB→AC
3. $\neg\neg A \rightarrow A$
4. $\rightarrow \forall x.A[A]_c^x$ where x
5. $\rightarrow \forall x. \rightarrow AB \rightarrow A\forall x.B$ where x is not free in A.

Inference Rules

1. (*modus ponens*) from A and →AB infer B
2. (*generalization*) from A, if x is a variable, infer $\forall x.A$.

Exercises

1. Rewrite the axioms in infix form.

5.2.2 Hilbert's Axiomatization

Axioms

1. $\rightarrow A \rightarrow BA$
2. $\rightarrow\rightarrow A \rightarrow BC \rightarrow\rightarrow AB \rightarrow AC$
3. $\rightarrow \wedge ABA$
4. $\rightarrow \wedge ABB$
5. $\rightarrow A \rightarrow B \wedge AB$
6. $\rightarrow A \vee AB$
7. $\rightarrow B \vee AB$
8. $\rightarrow\rightarrow AC \rightarrow\rightarrow BC \rightarrow \vee ABC$
9. $\rightarrow\leftrightarrow AB \rightarrow AB$
10. $\rightarrow\leftrightarrow AB \rightarrow BA$
11. $\rightarrow\rightarrow AB \rightarrow\rightarrow BA \leftrightarrow AB$
12. $\rightarrow\rightarrow \neg A\neg B \rightarrow BA$
13. $\rightarrow \forall x.A[A]_c^x$ where x
14. $\rightarrow \forall x. \rightarrow AB \rightarrow A\forall x.B$ where x is not free in A.

Inference Rules

1. (*modus ponens*) from A and $\rightarrow AB$ infer B

2. (*generalization*) from A, if x is a variable, infer $\forall x.A$.

Exercises

1. Rewrite the axioms in infix form.

Exercises

1. Rewrite the axioms in infix form.

5.3 Hilbert Style Proofs

The Hilbert style of proofs is often used in teaching geometry in high school. A Hilbert style system consists of a set of axioms and rules of inference. Proofs consist of the theorem to be proved followed by a sequence of lines each of which contains a theorem, assumption, or an axiom and a reason why it is a theorem with the last line the theorem being proved. Subproofs may be indented.

Hilbert Style Proof

Theorem to be proved: $A \rightarrow B$

Steps	**Reasons**
1.	1.
(a)	(a)
(b)	(b)
2.	2.

Each step consists of a formula. The corresponding reason is either assumption, instance of a theorem, or an inference rule. The inference rules are those of natural deduction. The point of a proof is to provide convincing evidence of the correctness of some statement. The following proof formats make clear the intent of the proof as it is read from beginning to end.

Natural Deduction Rule	Hilbert Style Proof Format	
$\dfrac{P, P \to Q}{Q}$	Q 1 P 2 $P \to Q$	By Modus Ponens *explanation* *explanation*
$\dfrac{A \vdash B}{A \to B}$	$A \to B$ 1 $\neg B$ 2 $\neg A$ 3 A	By Contrapositive Assumption *explanation* But A holds because *explanation*
$\dfrac{P, Q \vdash R}{P \wedge Q \to R}$	$P \wedge Q \to R$ 1 P 2 Q 3 R	By Deduction Assumption Assumption *explanation*
$\dfrac{\neg P \vdash Q \wedge \neg Q}{P}$	P 1 $\neg P$ 2 $Q \wedge \neg Q$	By Contradiction Assumption *explanation*
$\dfrac{P \vdash Q \wedge \neg Q}{P}$	$\neg P$ 1 P 2 $Q \wedge \neg Q$	By Contradiction Assumption *explanation*
$\dfrac{P \vee Q, P \to R, Q \to R}{R}$	R 1 $P \vee Q$ 2 $P \to R$ 3 $Q \to R$	By Case Analysis *explanation* *explanation* *explanation*
$\dfrac{P \to Q, Q \to P}{P \leftrightarrow R}$	$P \leftrightarrow Q$ 1 $P \to Q$ 2 $Q \to P$	By Mutual implication *explanation* *explanation*
$\dfrac{P(0), P(n) \to P(n+1)}{\forall n.P}$	$\forall n.P$ 1 $P(0)$ 2 $P(n)$ 3 $P(n+1)$	By Induction *explanation* (Base step) Assumption (Induction hypothesis) *explanation* (Induction step)

5.4 Natural Deduction

Natural deduction was invented independently by S. Jaskowski in 1934 and G. Gentzen in 1935. It is an approach to proof using rules that are designed to mirror human patterns of reasoning. There are no logical axioms, only inference rules. For each logical connective, there are two kinds of inference rules, an introduction rule and an elimination rule.

- Each *introduction rule* answers the question, under what conditions can the connective be introduced.
- Each *elimination rule* answers the question, underheat conditions can the connective be eliminated.

The natural deduction rules of inference are listed in Figure 5.2.

	Introduction Rules	**Elimination Rules**
$\neg$	$\dfrac{A \vdash B \wedge \neg B}{\neg A}$	$\dfrac{\neg A \vdash B \wedge \neg B}{A}$
$\wedge$	$\dfrac{A, B}{A \wedge B}$	$\dfrac{A \wedge B}{A, B}$
$\vee$	$\dfrac{A}{A \vee B}$	$\dfrac{A \vee B}{A}$
$\rightarrow$	$\dfrac{A \vdash B}{A \rightarrow B}$	$\dfrac{A, A \rightarrow B}{B}$
$\forall x.$	$\dfrac{P(x)}{\forall x.P(x)}$	$\dfrac{\forall x.P(x)}{[P(x)]^c_x}$ for any $c \in \mathcal{C}$
$\exists x.$	$\dfrac{P(c)}{\exists x.[P(x)]^x_c}$	$\dfrac{\exists x.P(x)}{[P(x)]^c_x}$ for new $c \in \mathcal{C}$

Figure 5.2: **Natural Deduction Inference Rules**

The nature of many proofs in natural deduction consists of picking apart a logical expression using the elimination rules to get at the constituent parts and then building up new expressions from the constituent parts using the introduction rules. Natural deduction inference rules may be used in Hilbert style proofs and in sequent systems.

5.5 The Analytic Properties

Analytic properties of formulas refer to the logical meaning of formula. The method takes formulas apart and searches for contradictions among the resulting sub-formulas. Thus analytic methods are associated with refutation style theorem proving. The compound formulas (with the exception of the negation of an atomic formula) are classified as of type α with sub-formulas α_1 and α_2,type β with sub-formulas β_1 and β_2, and type γ, or of type δ. The classification scheme for formulas of classical first-order logic is summarized in Figure 5.3. The classification can also be applied to modal logics. Analytic methods are utilized the tableaux method and in sequent systems. Figure 5.3 lists the analytical properties of the classical logical connectives.

The classification of the modal operators depends on the underlying model.

Definition 5.1 *By a* **Hintikka (downward saturated) set** *we mean a set S such that the following conditions hold for every formula of type alpha, beta, gamma, and delta in S.*

1. *No* atomic *formula and its negation are both in S.*
2. *If alpha is in S, then both alpha*$_1$ *and alpha*$_2$ *are in S.*
3. *If beta is in S, then either beta*$_1$ *is in S or* β_2 *is in S.*
4. *If* γ *is in S, then for every c,* $\gamma(c)$ *is in S.*
5. *If* δ *is in S, then for some d,* $\delta(d)$ *is in S.*

Downward saturated sets are guaranteed to be coherent and consistent. The construction of downward saturated sets is a purely syntactic procedure which produces a semantic truth assignment (truth function) for the set.

Lemma 5.1 Hintikka's lemma for first-order logic *Every Hintikka set S is satisfiable.*
Proof: *A valuation function is easily constructed from the Hintikka set. The valuation function maps all atomic formula S to t and those not appearing in the set to f. The construction rules follow the rules for satisfiability. QED.*

And	alpha	alpha_1	alpha_2
	$A \wedge B$	A	B
	$\Box(A \wedge B)$	$\Box A$	$\Box B$
	$\forall x.(A \wedge B)$	$\forall x.A$	$\forall x.B$
Or	beta	beta_1	beta_2
	$A \vee B$	A	B
	$\Diamond(A \vee B)$	$\Diamond A$	$\Diamond B$
	$\exists x.(A \vee B)$	$\exists x.A$	$\exists x.B$
Universal	gamma	gamma(c)	
	$\forall x.A$	$[A]_x^c$	Any $c \in \mathcal{C}$
Existential	delta	delta(d)	
	$\exists x.A$	$[A]_x^d$	d is new

Equivalences:

Negation
$\neg\neg A \equiv A$
$\neg(A \vee B) \equiv (\neg A \vee \neg B)$
$\neg(A \wedge B) \equiv (\neg A \wedge \neg B)$
$\neg\Box A \equiv \Diamond\neg A$
$\neg\Diamond A \equiv \Diamond\neg A$
$\neg\forall x.A \equiv \exists x.\neg A$
$\neg\exists x.A \equiv \forall x.\neg A$
Distributive Properties
$A \wedge (B \vee C) \equiv (A \wedge B) \vee (A \wedge C)$
$\exists x.(A \vee B) \equiv (\exists x.A \vee \exists x.B)$
$\forall x.(A \wedge B) \equiv (\forall x.A \wedge \forall x.B)$
Commutative Properties
$(A \vee B) \equiv (B \vee A)$
$(A \wedge B) \equiv (B \wedge A)$
$\forall x.\forall y.A \equiv \forall y.\forall x A$
Other
$(A \rightarrow B) \equiv (\neg A \vee B)$
$(A \leftrightarrow B) \equiv (A \wedge B) \vee (\neg A \wedge \neg B)$

Figure 5.3: Analytic Subformula Classification

5.6 The Method of Analytic Tableaux

The method of analytic tableaux builds a proof tree using the analytic properties (Section 5.5) of formulas which involves replacing a compound formula with one or more sub-formulas. The the proof terminates when a contradiction is found. Thus, like resolution, the method is based on refutation but is interesting because it builds a model of the formula under proof.

Tableau Construction

The *tableau method* is a *backward-chaining* proof search method. The tableau is a tree with sets of formulas (a block) at each node and leaf. The construction begins with a set of formulas placed at the root of the tree (the negation of the theorem to be proved is included in the set of formulas). The tree is extended by adding a new block as required by one of four reduction rules. The construction of a branch is terminated when a contradictory block is constructed or when no reduction rule applies. The construction of the tree is terminated when all branches are terminated.

We use the following conventions:

- p, q denote atomic formulas
- P, Q, and R denote formulas
- X, Y, and Z denote sets of formulas
- X, Y stands for $X \cup Y$ and X, P stands for $X \cup \{P\}$
- *Lit* stands for a set of literal formulas - atomic formulas and negations of atomic formulas.

In addition, we assume (though it is not necessary) that formulas are in negation normal form.The form of the tableau rules for extending a branch, creating a new branch, and terminating a branch are given in Figure 5.4.

Each reduction rule corresponds to one of the analytic properties (Section 5.5). Given a block of formulas containing a formula of type α, β, γ, δ, the reduction rules specify the replacement of a block with one or more blocks in which the formula is replaced with its sub-formulas. For example, *Rule A* permits the replacement of a conjunction with the conjuncts and *Rule B* requires the block to be replaced with two blocks each containing one of the disjuncts.

By a *block tableau* for a finite set, *Fs*, of formulas, we mean a tree constructed by placing the set *Fs* at the root, and then continuing according to the block tableau inference rules in Figure 5.5.

Definition:

Linear Extension: Current Block

Child Block

Branching Extension: Current Block

Left Branch Right Branch

Branch Termination: *Lit* or Current Block contains p, $\neg p$

Figure 5.4: Block Tableau Construction

Rule A: $\boxed{\text{S}, \alpha}$

$\boxed{\text{S}, \alpha_1, \alpha_2}$

Rule B: $\boxed{\text{S}, \beta}$

$\boxed{\text{S}, \beta_1}$ $\boxed{\text{S}, \beta_2}$

Rule C: $\boxed{\text{S}, \gamma}$

$\boxed{\text{S}, \gamma(c), \gamma}$ for any $c \in \mathcal{C}$

Rule D: $\boxed{\text{S}, \delta}$

$\boxed{\text{S}, \delta(c)}$ where $c \in \mathcal{C}$ is new to the tree

Figure 5.5: Block Tableau Inference Rules

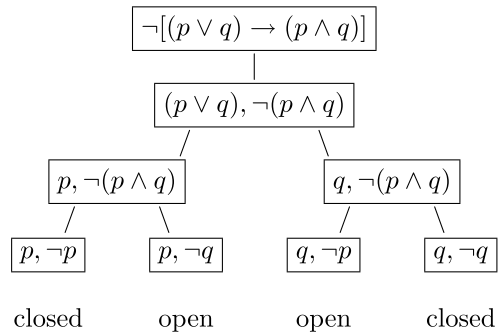

Figure 5.6: Tableau for $\neg[(p \vee q) \to (p \wedge q)]$

- A *path* in tableau is *closed/contradictory* if *a block on the path contains a formula and its negation.*
- A *path* in tableau is *open* if no block on the path contains a formula and its negation.
- A *tableau* is *contradictory* if *every path is contradictory.*
- A *proof* of A from a set of formulas Ss, $Ss \vdash A$, is a contradictory tableau rooted at $\boxed{Ss, \neg A}$.

Figure 5.6 is a tableau for $\neg[(p \vee q) \to (p \wedge q)]$. The open blocks provide a model for the formula.

Figure 5.7 is a tableau proof of $\forall x.[P(x) \to Q(x)] \to [\forall x.P(x) \to \forall x.Q(x)]$.

Since all branches of the tableau are closed, the formula is proved.

For efficiency, apply the rules in the following order:

- rule A,
- rule C (but do not reuse a formula until other rules have been applied),
- rule D,
- rule B, and
- place used gamma formulas last in a list of formulas to be used.

Model Construction

Classical propositional logic has the finite model property–there is a finite set of finite sets of atomic formulas which determine the truth value of a formula. For example the formula $\neg a \vee b$ is true in either of the two sets in $\{\{\neg a\}, \{b\}\}$.

$$\boxed{\neg(\forall x.[P(x) \to Q(x)] \to [\forall x.P(x) \to \forall x.Q(x)])}$$

$$\boxed{\forall x.[P(x) \to Q(x)], \neg[\forall x.P(x) \to \forall x.Q(x)]}$$

$$\boxed{\forall x.[P(x) \to Q(x)], \forall x.P(x), \neg\forall x.Q(x)}$$

$$\boxed{\forall x.[P(x) \to Q(x)], \forall x.P(x), \neg Q(a)}$$

$$\boxed{\forall x.[P(x) \to Q(x)], P(a), \neg Q(a), \forall x.P(x)}$$

$$\boxed{P(a) \to Q(a), P(a), \neg Q(a), \forall x.P(x), \forall x.[P(x) \to Q(x)]}$$

$$\begin{array}{cc} \boxed{\neg P(a), P(a), \neg Q(a), \forall x.P(x), \forall x.[P(x) \to Q(x)]} & \boxed{Q(a), P(a), \neg Q(a), \forall x.P(x), \forall x.[P(x) \to Q(x)]} \\ \text{closed} & \text{closed} \end{array}$$

Figure 5.7: Tableau for $\forall x.[P(x) \to Q(x)] \to [\forall x.P(x) \to \forall x.Q(x)]$

The tableau method can be used to construct these models. If all branches in the tableau are contradictory, the formula is unsatisfiable and any open branch is a model of the formula.

Implementations of the tableau method for classical propositional logic and one for propositional modal logic is available.

5.7 Sequent Systems (Gentzen)

A *sequent* is a pair of sets of formulas separated by the turnstile,

$$[U \vdash V].$$

Alternative notations include $[U \to V]$ and $[U \Rightarrow V]$. The first set, U, is referred to as the *antecedent* of the sequent and the second set, V is called its *succeedent*. A sequent corresponds to the assertion that if every formula in U holds, then some formula in V holds. Symbolically,

$$A_1 \wedge ... \wedge A_m \to S_1 \vee ... \vee S_n.$$

Sequent systems have many inference rules and one logical axiom. The single logical axiom is:

$$[U, A \vdash V, A].$$

Initial sequent

$$[U \vdash T]$$

Axiom Final sequent

$$[U, X \vdash V, X]$$

	Left Rule	**Right Rule**	
Negation	$\dfrac{[U, \neg F \vdash V]}{[U \vdash V, F]}$	$\dfrac{[U \vdash V, \neg F]}{[U, F \vdash V]}$	
Rule A	$\dfrac{[U, \alpha \vdash V]}{[U, \alpha_1, \alpha_2 \vdash V]}$	$\dfrac{[U \vdash V, \beta]}{[U \vdash V, \beta_1, \beta_2]}$	
Rule B	$\dfrac{[U, \beta \vdash V]}{[U, \beta_1 \vdash V], [U, \beta_2 \vdash V]}$	$\dfrac{[U \vdash V, \alpha]}{[U \vdash V, \alpha_1], [U \vdash V, \alpha_2]}$	
Rule C	$\dfrac{[U, \gamma \vdash V]}{[U, \gamma, \gamma(c) \vdash V]}$	$\dfrac{[U \vdash V, \delta]}{[U \vdash V, \delta, \delta(c)]}$	Any $c \in C$
Rule D	$\dfrac{[U, \delta \vdash V]}{[U, \delta(c) \vdash V]}$	$\dfrac{[U \vdash V, \gamma]}{[U \vdash V, \gamma(c)]}$	Some $c \in C$ new to the sequent

Figure 5.8: Analytic Sequent Inference Rules

The inference rules based on the analytic properties of formulas are given in Figure 5.8.

A formula is a theorem if it is possible to infer an instance of the axiom. A proof consists of constructing a finite tree of sequents using inference rules based on the analytic properties of formulas and natural deduction rules. Each sequents follows from the immediately preceding sequent by an application of an inference rule. At the root of the tree is the sequent

$$[\text{Axioms, and previously proved theorems } \vdash \text{Theorem to be proved}].$$

The tree is constructed by the application of the inference rules (Figure 5.8). The proof ends if each branch ends with the sequent at the leaf of the form

$$[U, A \vdash V, A].$$

Proofs using theories (a theory is a set of formulas) are implemented in sequents by placing the theory on the left and the formula to be proved on the right,

$$[\text{Theory} \vdash \text{Formula}].$$

The inference rules may be used to construct either direct or indirect proofs.
Direct proof: To prove $[U \vdash T]$, use the rules breakdown and reassemble the formulas on the left until $[U, T \vdash T]$ is derived.
Goal oriented proofs: To prove $[U \vdash B]$, use both left and right rules to breakdown and assemble formulas until an instance of the axiom occurs on all branches.

Different sequent systems are characterized by the set of inference rules and axioms.

Example

$$
\begin{array}{c}
\text{Proof of } [(A \wedge B) \to C \vdash A \to (B \to C)] \\ \hline
[(A \wedge B) \to C, A \vdash (B \to C)] \\ \hline
[(A \wedge B) \to C, A, B \vdash C] \\ \hline
\begin{array}{c|c}
\begin{array}{c} [C, A, B \vdash C] \\ \text{closed} \end{array} &
\begin{array}{c} [A, B \vdash A \wedge B, C] \\ \hline \begin{array}{c|c} [A, B \vdash A, C] & [A, B \vdash B, C] \\ \text{closed} & \text{closed} \end{array} \end{array}
\end{array}
\end{array}
$$

An implementation for classical propositional logic is available (`code/propseq`). An implementation for classical first-order logic is available (`code/folseq`).

Bibliography

[1] Beckert, Bernhard and Goré, Rajeev *ModLeanTAP* `i12www.ira.uka.de/modlean`

[2] Beckert, Bernhard and Posegga, Joachim *LeanTAP* `i12www.ira.uka.de/~posegga/leantap/leantap.html`

[3] Fitting, Melvin

[4] Otten, Jen *ileanTAP* `aida.intellektik.informatik.th-darmstadt.de/~jeotten/ileanTAP`

[5] Smullyan, Raymond M. *First-Order Logic* Springer-Verlag New York Inc. 1968.

[6] XRefer `http://www.xrefer.com/entry/552896` Natural Deduction

Chapter 6

Semantics

6.1 Introduction

An atomic formula is f said to be *satisfiable* if there is a total function on the set of atomic formulas to some satisfaction condition which satisfies f. Given a function, a formula is said to be *valid* for that function if it is satisfied by the function. For most logics, the satisfaction condition of interest is whether or not the formula is true.

$$v : \mathcal{P} \to \{\text{false}, \text{true}\}$$

In general, a formula A is *satisfiable*, $\mathcal{M} \models A$, if it is satisfied in the model $\mathcal{M}$. Satisfiability is defined in terms of the satisfiability of atomic formulas and the meaning of the logical operators. We say that $\mathcal{M}$ models the formula A ($\mathcal{M}$ is a model for A) and write $\mathcal{M} \models A$ if $A \neq \mathbf{f}$ and the following hold:

- if $A \in \mathcal{P}$ and $v(A)$ is satisfied.
- if A is $\neg B$ and not $M \models B$
- if A is $\to BC$ and either $M \models C$ or not $M \models B$
- if A is $\forall x.B$ and for all constants, t, $M \models [B]_x^t$

A formula is said to be a *contradiction* if it is not satisfiable (or it has no model). The formula $\mathbf{f}$ is a contradiction. A formula S is *valid*, $\models S$, if it is satisfied in all models $\mathcal{M}$. Valid formulas are also called *tautologies*.

A set of formulas Fs, is *satisfiable*, $\mathcal{M} \models Fs$, if there is a structure $\mathcal{M}$ in which all of the members of Fs are satisfied. Such a structure is called a *model* of Fs. If Fs has no model, it is said to be $unsatisfiable$. The set of formulas $\{A, \neg A\}$ is unsatisfiable.

A formula A is a *logical consequence* of a set of formulas Fs, $Fs \models A$, if every model of the formulas Fs satisfies A i.e., for every $\mathcal{M}$ such that $\mathcal{M} \models Fs$, $\mathcal{M} \models A$.

Move this!

The semantical methods for programming languages include these concepts.

- If S is a language and a subset of L, then the semantics are called *reduction semantics.*

- If S is a language different from L, then the semantics are called *translation semantics.*

- If S is a mathematical object, then the semantics are called *denotational semantics.*

- If **S** is a sequence of operations or actions, then the semantics are called *operational semantics.*

6.2 Semantic Approaches

In logic, there are three equivalent approaches to semantics, Herbrand semantics which let formulas stand for themselves, valuation functions which identify a formula with a truth value, and Tarskian semantics which interpret formulas in terms of correspondence with objects in a structure. This section presents a variety of ways of defining satisfaction conditions.

Herbrand Semantics

A way of characterizing the meaning of a formula or theory that is independent of how we might intend to interpret its symbols is to let ground expressions (expressions without variables) stand for themselves. This was Herbrand's insight. Herbrand semantics are important in relational databases and logic programming and are at the heart of literal approaches to literary interpretation.

Hebrand semantics for a set of formulas is defined as follows:

Let Fs be a subset of $\mathbf{F}$ where $\mathcal{L} = \langle \mathcal{F}, \mathcal{P}, \mathcal{V}, \mathbf{F} \rangle$
Let C be the constants appearing in Fs.
Let F be the functors appearing in Fs.
$H_T \subset \mathcal{F}$ only containg symbols in $C \cup F$ the *Herbrand Universe* of Fs.
$H_B \subset \mathcal{P}$ whose terms are in H_T is the *Herbrand Base*
$\mathcal{M}_H \subset H_B$ is the *Herbrand Model*

$\mathcal{M}_H \models F$ (F is a formula in $\mathcal{L}$) is defined by:

not $\mathcal{M}_H \models \mathbf{f}$
$\mathcal{M}_H \models p$ iff $p \in \mathcal{M}_H$
$\mathcal{M}_H \models \rightarrow AB$ iff not $\mathcal{M}_H \models A$ or $\mathcal{M}_H \models B$
$\mathcal{M}_H \models \forall x.A$ iff $\mathcal{M}_H \models [A]^t_x$ for all $t \in H_T$

The *Herbrand Universe* of T is the set H_T of all ground terms that can be formed from the constants and function symbols in T.

The *Herbrand Base* of T is the set B_T of atomic formulas formed using the Herbrand universe H_T and and the predicate symbols P_T of T.

A *Herbrand Model* is an assignment of truth values to the elements of B_T.

A *Herbrand Interpretation* is an interpretation where the domain is a Herbrand universe and where the interpretation function interprets each constant and function symbol as 'itself'.

A Herbrand model for a set of formulas T is a Herbrand interpretation that satisfies T.

Theorem 6.1 *The intersection of two Herbrand models is a model.*

Proof: This is proved quite easily. Suppose that I and J are models of a set of formulas T. Consider $M = I \cap J$. Certainly M is a subset of the Herbrand base, so M is an interpretation. To show that M is a model we must show that for each instance of a clause in the program $A \rightarrow B_0, ..., B_n$, A is in M if $B_0, ..., B_n$ are in M. Pick any instance of any clause of the program; if all the goals in the body of the clause are in M, then they are also in both I and J. This means that A is in both I and J (since they both are models). But then A is in the intersection of I and J, which is M. rewrite this proof

We define the *minimal model* of a theory T, M(T), to be the intersection of all models. M(T) is the declarative meaning of T.

Boolean Semantics and Valuation Functions

In classical logic there are precisely two *truth values*: *true* and *false* . In this formalization, the meaning of a formula determined by a *valuation function* v

which maps atomic formulas to truth values. A valuation function v satisfies the following:

$$v(\mathbf{f}) = 0$$
$$v(p) = \begin{cases} 0 & \text{if } p \text{ is false} \\ 1 & \text{if } p \text{ is true} \end{cases} \quad \text{where } p \in \mathcal{P}.$$

Multivalued logics (see Chapter 9) are logics whose valuation function may have more than two values. Fuzzy and probabilistic logics are examples of continuous valued logics which have truth values in the closed interval [0, 1]. The exercises suggest some valuation functions for various types of logics.

Truth Tables?
Boolean semantics are based on the coherence theory of truth.

For propositional formulas, truth tables are an accepted method to determine whether a formula is a tautology (valid), satisfiable, or a contradiction.

Tarskian Semantics

A relational structure is a set with relations. Often in mathematics, science, and engineering, a relational structure is the object of interest and a language is created in order to say things about it. Tarski's insight was to see that a way of characterizing the meaning of a formula or theory is to show how to interpret its symbols in some relational structure.

$\mathcal{S} = \langle \mathcal{C}, \mathcal{R} \rangle$ is a simple relational structure where

$\mathcal{C} = \{c_0, c_1, ...\}$ – A possibly infinite set of constants.
$\mathcal{R} = \{R^i_j \mid R^i_j \subset \mathcal{C}^i\}$ – A possibly infinite set of relations

$\mathcal{L} = \langle \mathcal{C}, \mathcal{P}, \mathcal{V}, \mathbf{F} \rangle$ is a language for the relational structure $S = \langle C_s, R_s \rangle$ where

$\mathcal{C} = \{c_0, c_1 ...\}$ – A set of constants corresponding to C_s
$\mathcal{P} = \{P^i_j t_1 ... t_i \mid t_k \in \mathcal{F},\ i, j \in \mathbb{N}\}$ – A set of predicates corresponding to R_s
$\mathcal{V} = \{x_i \mid i \in \mathbb{N}\}$ The set of variables
$\mathbf{F} ::= \mathbf{f} \mid \mathcal{P} \mid\ \rightarrow \mathbf{FF} \mid \forall x.[\mathbf{F}]^x_t \mid \Box\mathbf{F}$ – The set of formulas where t, t_i are meta variables for functions, x is a variable for variables, and textual substitution, $[F]^t_x$ is a part of the meta language. Substitution means, replace each occurrence of t with x.

$\mathcal{I}$ is the interpretation of the elements of $C \cup P$ in $C_s \cup R_s$. It is a mapping between the ground terms, and atoms of the language L and the constants and relations in the structure S. The constants in a language are names of the constants in the structure and predicate functors are the names of relations. The interpretation function maps atomic formulas an appropriate tuple in the

structure. The interpretation function provides the satisfaction condition. The structure, language, and interpretation are combined to model the formulas of the language.

Given a model $\mathcal{M} = \langle \mathcal{L}, \mathcal{S}, \mathcal{I} \rangle$ where

$\mathcal{L}$ is a language,
$\mathcal{S} = \langle \mathcal{C}, \mathcal{R} \rangle$ is a relational structure,
$\mathcal{I} : \mathcal{L} \to \mathcal{S}$ is an interpretation and
$\mathcal{M} \models F$ (F is a formula in $\mathcal{L}$) is defined by:

not $\mathcal{M} \models \mathbf{f}$	
$\mathcal{M} \models p$	iff $p \in \mathcal{P}$ and $\mathcal{I}(p) \in \mathcal{R}$
$\mathcal{M} \models \to AB$	iff not $\mathcal{M} \models A$ or $\mathcal{M} \models B$
$\mathcal{M} \models \forall x.A$	iff $\mathcal{M} \models [A]_x^t$ for all $t \in \mathcal{F}$

For more complex situations, Etchemendy has suggested an extension of Tarskian semantics called *Representational Semantics*. In representational semantics, a sentence σ is true in model A if and only if σ would be true if the world were as depicted by A, that is, if A were an *accurate* model. The accuracy of models is important for scientific theories and engineering. This and related issues are discussed in Chapter 7.

Kripke semantics

Kripke semantics are also called possible world semantics. In a Kripke model $M_K = \langle W, A \rangle$ there is a set of worlds W (each of which may be represented as a valuation function, relational structure, or a Herbrand model) and an accessibility relation A which describes whether or not a world is accessible from another world. Kripke semantics are used in modal logics (See Chapter 11).

Linear time temporal logic is an example of a logic that uses multiple world semantics. Each time increment is represented by a world. The accessibility relation is reflexive and transitive but not symmetric as we assume that time does not run backwards. For the formula $\Box A$, A holds in the current world and in all future worlds and for the formula $\Diamond A$, A holds in either the current world or some future world.

Let L be the language of first-order logic augmented with modal operators $\Box$ and $\Diamond$, T be a set of formulas of L, A_T, F_T, and P_T be, respectively, the sets of constants, function symbols, and predicate symbols appearing in the formulas of T. A *ground term* (respectively, *atom*, *literal*, or *sentence*) is a term (respectively, atom, literal, or sentence) which has no variables.

Multiple structures are used to model *modal* formulas. A *model of a theory* is a structure and an interpretation that satisfies the formulas of the theory.

A model $M = (W,\ A)$ is a Kripke structure. A Kripke structure where $|\mathbf{W}| =$

$\mathcal{U} = \langle \mathcal{W}, \mathcal{A} \rangle$ – a multiworld universe

$\mathcal{W} = \{R \mid R \text{ is a relational structure}\}$ – a set of worlds.
$\mathcal{A} \subset \mathcal{W} \times \mathcal{W}$ – an accessability relation between worlds.

Figure 6.1: Multiple world structure $\mathcal{U} = \langle W, A \rangle$

1 corresponds to traditional logics.
A reachability relation that is symmetric (**Axy** implies **Ayx**) implies that the graph is nondirectional.
A reachability relation that is transitive (**Axy** and **Ayz** implies **Axz**) can be used to model temporal phenomenon.
A reachability relation that is reflexive (**Axx**), symmetric, and transitive, can be used to reason about finite state systems.

$M_K = \langle W, A \rangle$ and $u \in W$ is a model of the theory T if for any closed sentence ϕ of T M_K satisfies ϕ, denoted by $M_K \models_u \phi$ where

- ϕ is an atomic formula and $u \models \phi$
- ϕ is $\alpha \supset \beta$ and either $M_K \not\models_u \alpha$ or $M_K \models_u \beta$
- ϕ is $\neg\alpha$ and $M_K \not\models_u \alpha$
- ϕ is $\forall x \alpha(x)$ and for each $t \in Tr$, $M_K \models_u \alpha(t)$
- ϕ is $\Box\alpha$ and for each $v \in W$, such that Auv and $M_K \models_v \alpha$ and
- ϕ is $\Diamond\alpha$ and there is a $v \in W$ such that Auv and $M_K \models_v \alpha$.

Note: Read $M_K \models_u \phi$ as ϕ holds in M_K at u or M_K satisfies ϕ at u.

6.3 Model Theory

In model theory, relational structures are the objects of primary interest. Language is introduced in order to discuss the properties of the structure. Thus, Thus, model theory begins with a class of set-theoretic objects called *structures*, *interpretations*, or *models* and constructs a mapping from some language to the objects.

Formal systems involve a correspondence between a language and a structure. In proof theory, we regard the language as the object of primary interest. For most of us, the language is the most visible. However in many cases, the natural approach is to begin with the structure.

Relationships between structures

$\mathcal{M} = \langle W, A \rangle$ is a graph where
$W = \{w \mid w$ is a relational structure$\}$
$A \subset W \times W$ – is the accessibility relation

$M \models \bot$	iff $v(\bot) =$ false
$M \models \top$	iff $v(\top) =$ true
$M \models p$	iff $v(p) \in w$
$M \models \neg A$	iff not $M \models A$
$M \models (A \vee B)$	iff $M \models A$ or $M \models B$
$M \models (A \wedge B)$	iff $M \models A$ and $M \models B$
$M \models A \rightarrow B$	iff $M \models \neg A$ or $M \models B$
$M \models \neg(A \rightarrow B)$	iff $M \models A$ or $M \models \neg B$
$M \models \Box A$	iff $M' \models A$ for all u such that Awu and $M' = (U, u, v)$
$M \models \Diamond A$	iff $M' \models A$ for some u such that Awu and $M' = (U, u, v)$
$M \models \forall x F$	iff $M \models [F]^c_x$ for all $c \in C$
$M \models \neg\forall x F$	iff $M \models \neg[F]^c_x$ for some $c \in C$

Figure 6.2: $M_K \models_u F$ where $M_K = (W, A)$

- Substructure (subrealization) – extension
- Restriction
- Embedding
- Isomorphic
- Elementarily equivalent

A set of sentences is *consistent* iff it has a model. A *proof method* is *complete* iff *every valid formula has a proof.* A proof method is *incomplete* iff there is a valid formula for which there can be no proof.

Theorem 6.2 *For infinite structures following two statements are false:*

1. *Every property expressed in a language defines a subset of a relational structure.*
2. *Every subset of a relational structure has a defining property in a language.*

Proof: The first statement is false because the paradoxes of set theory provide contradictions demonstrating that the statement is false (e.g. set of all sets). The second statement is false (Skolem 1929) because

Syntax	**Semantics**	
	Semantic Mapping	*Relational Structure*
Formulas e.g., $F ::= \mathbf{f} \mid P \mid \to FF \mid \forall x.F$ **Theorems** non-Logical Axioms of Theory T Tautologies Logical Axioms	$\to$	A set of constants and a set of relations defined on the set of constants. 1. A nonempty set A called the universe. 2. A set of n-ary functions from A^n to A, n=0, 1, ... 3. A set of n-ary relations in A^n, n=0, 1, 2, ... In pure logic the relational structure is the set {0, 1} where 0 is interpreted as false and 1 as true. The semantic mapping is called a *valuation function* which is a total function from the set of atomic formulas to the set {0, 1}
Inference rules e.g., modus ponens **Modus ponens** $\dfrac{\text{From: } A, A \to B}{\text{Infer: } B}$		

Figure 6.3: Formal System

1. Assume a countably infinite relational structure.
2. The power set of a countable infinite set is uncountable.
3. There are an uncountable number of sets, if each set has a defining property, then the set of defining properties is also uncountable.
4. For a given language, each property is expressed using a finite string. The set of such strings is countably infinite.
5. Hence there are sets without defining properties.

Theorem 6.3 The Löwenheim-Skolem Theorem: *Let Σ be a set of sentences with an infinite model. If Σ is finite, Σ has a model of each infinite cardinal. If Σ is infinite and of cardinal α then Σ has a model of any cardinal $\geq \alpha$.*

Theorem 6.4 The compactness theorem for the predicate calculus: *A set Σ of sentences of a language L has a model iff each finite subset of Σ has a model.*

Theorem 6.5 The Gödel-Henkin completeness theorem: *A set Σ of sentences has a model iff it is consistent.*

The sentence X of language L **follows logically** from the sentences of the class K if and only if every model of the class $\mathbf{K}$ is a model of the sentence X (Tarski).

Let A be a structure, L a language, and σ be a sentence in the language. If the sentence σ is *true in the structure* A we write $A \models \sigma$. A sentence is said to be logically true if it is true in all structures. A sentence σ is said to be a logical consequence of a set Σ of sentences if σ is true in every structure in which the members of Σ are all true.

A formula f is said to be true if $M_S(f) = \mathit{True}$. A formula$f$ is said to be false if $M_S(f) = \mathit{False}$.

A formula f is said to be *satisfiable* if $M_S(f) = \mathit{True}$ (also written $M_S \models f$). Example: A. We write $T \models f$ to say that f is satisfiable whenever the formulas in the theory T are satisfiable.

A formula f is said to be *valid* (also a *tautology*) if $M_S(f) = \mathit{True}$ for all structures and validation functions (also written $\models f$). Example: $A \vee \neg A$.

A formula f is said to be a *contradiction* (or *falsehood*) if under every validation function,f is unsatisfiable (false). Example: $A \wedge \neg A$

Two formulas f and g are said to be *inconsistent* if under every validation function, one is true and the other is false. Example: A, $\neg A$.

Two formulas f and g are said to be *independent* if there is a validation function under which both are true and a validation function under which one is true and the other is false. Example: $M_S \models A \wedge B$ and $M\prime_{S\prime} \models A \wedge \neg B$.

6.4 References

Bell, J. L. and Slomson, A. B. Models and Ultraproducts: an introduction. North-Holland Publishing Company. 1969.

Etchemendy

Stanford Encyclopedia of Philosophy "Model Theory" `http://plato.stanford.edu/entries/model-theory`

6.5 Exercises

1. For each of the following valuation functions, find a way to extend it to all formulas and show that your extension satisfies the models relation.
   (a) $v : \mathcal{P} \to [0, 1]$ is a valuation function for a two valued .
   (b) $v : \mathcal{P} \to [0, 1, ..., n]$ is a valuation function for a multivalued logic.
   (c) $v : \mathcal{P} \to [0, 1]$ is a valuation function for a continuous, probalistic, or fuzzy logic.
   (d) $v : \mathcal{P} \to [\perp, 0, 1]$ is a valuation function for a logic where $v(f) = \perp$ if the truth value of f is unknown.
2. Use truth tables to justify the abbreviations and other logical equivalences.
3. Show that Boolean and Tarskian semantics are equivalent.

Chapter 7

Relational Structures

7.1 Overview

As used in science, a theory or model (theory used in what follows) is a symbolic representation of a relational structure. A relational structure is a set of objects, their relationships. The relational structure is usually an idealized representation of the actual structure (domain) of interest. Figure 7.1 illustrates the relationship between a theory, the structure it represents, and the actual domain of interest.

Because of this idealization, theories are caricatures. Just as in a caricature, where the characteristic features of the subject represented are distorted, exaggerated, ennobled or beautified for effect, theories attempt to reveal the hidden ideal of the phenomena they seek to explain. The construction of the idealized structure is governed by methods of particular disciplines. The natural and social sciences follow various experimental protocols. The humanities utilize various hermeneutical methods. The engineering disciplines utilize an array of methods. During the requirements phase of engineering design methods from the social sciences and the humanities are used while during the design phase, methods from the mathematical and physical sciences are employed.

While the relationship between the theory and the idealized structure is precise, the relationship of interest is often between the theory and the actual structure and this relationship is often vague. *Mathematical theories* are precise and accu-

$$\text{Theory} \quad \rightarrow \quad \text{Relational Structure} \quad \rightarrow \quad \text{Domain}$$

Figure 7.1: Theory, Relational Structure, and Domain

rate descriptions of the properties of a static, completely describable, universe. *Scientific theories* are vague (within a margin of error) descriptions of the invariant (recurring) properties of a complex, deterministic, dynamic structures of the natural world. Due to the complexity of the universe, the margins of error, and the relationship of the observer to the universe of study, the theories cannot be known to be correct or complete. However, they are required to be objective and verifiable within the margin of error. *Speculative or forensic scientific theories* are the application of standard scientific theories to reconstruct non-recurring (historical) events or predict potential future events. Examples include cosmology, geology, evolutionary theory, and forensic science. *Social science theories* are descriptions of non-recurring properties of a dynamic universe. *Humanistic theories* focus on how people view themselves and others.

... verifiability ... differing views of the value of particular idealizations ... objective ... subjective ... social group ...

7.2 Relational Structures

A formal system consists of a language, a structure, a correspondence between the language and the structure and some inference rules that describe how to construct a new sentence from zero or more sentences. Axioms, sentences in the language, are used to describe the properties of the structure and the inference rules support reasoning about the structure.

Formally, a relational structure is:

1. A set of constants and a set of relations defined on the set of constants.
2. If $G = \langle V, E \rangle$ is a graph with V a set of relational structures and E a set of edges, then G is a relational structure.
3. Only those structures that satisfy 1 and 2 are relational structures.

Let Σ be a possibly countably infinite set of symbols and L(Σ) be a language (a set of sentences or strings), composed of symbols from Σ. Let S be a relational structure and I a correspondence (semantic relation) between L(Σ) and S. Let M=$\langle S,\ I\rangle$ be called a *structure* for the language. For a sentence, ϕ, of the language, L(Σ), the structure M is called a *model* of ϕ, written $M \models \phi$, if the sentence is true in M i.e., $I(\phi,S)$ is true. A *theory* is a set of sentences and for a given theory T, $M \models T$ denotes that $M \models \phi$ for each sentence ϕ in T (T is said to be true in (about) structure M, if $M \models T$) . A sequence of sentences is a proof (of the last formula in the sequence) if each sentence is either in the theory T or is derived from one or more of the previous sentences in the sequence by a sound rule of inference. $T \vdash \phi$ denotes that the sentence ϕ has a proof from the sentences in T. Theories are *closed under deduction* i.e., if $T \vdash \phi$, then ϕ is in T. For more details see Irvine.

Examples of relational structures include

- Family database: set of individuals and the associated family relationships.
- Phone book: set of individuals, set of numbers, name and number relationship.
- Natural number arithmetic: set of numbers, order relation, arithmetic operations, etc.
- Turing machines
- The state diagram for a communication protocol.

7.3 The Relationship between the Relational Structure and the Domain

A semantic relation may be 1-to-1 giving an accurate representation. However, it is not always possible to construct accurate representations. For example, the word "red" is mapped to a number of actual colors and ambiguously to many others. Most mappings in scientific theories are vague. A vague mapping is 1-to-many. Bertrand Russell's paper on vagueness (in Chapter 30) is a highly readable discussion of these ideas. The following definitions are helpful in creating a language for semantic relations.

- The relation between the representing system A and the represented system B is *meaning*. The relation is
  - an*injection (1-to-1)* if each element of A maps to a unique element in B,
  - a *surjection (onto*) if for each element of B there is an element of A that maps to it,
  - *a bijection* if it is both 1-to-1 and onto,
  - *1-to-many* if some element of A maps to more than one element in B,
  - *partial* if some element of A does not map to an element of B,
  - *nondeterministic* if the mapping of some element of A to elements of B varies over time, and
  - other relationships include *speculative*, *probabilistic*, *fuzzy*, and *chaotic*.
  - *computable* if it is an injection and is a computable function,
  - *decidable* if there is an effective procedure for determining whether or not b=f(a).
  - Relations defined by rules of deduction are not, in general, decidable according to Gödel's incompleteness theorem.

- A statement in a representing system is
  - *vague* if many possible facts in the represented system could verify it and it is undecidable whether a fact verifies it,
  - *ambiguous*: if many possible facts in the represented system could verify it and it is decidable whether a fact verifies it,
  - *precise* if only one fact in the represented system would verify it ,
  - *valid* (*true*, corresponds to reality) if there is a corresponding fact in the represented system and
  - *accurate* if it is both precise and valid.
- A representing system is
  - *vague* if some statement in the representing system is vague (the semantic mapping is 1-to-many and undecidable),
  - *ambiguous* if some statement in the representing system is ambiguous (the semantic mapping is 1-to-many),
  - *precise* if each statement in the representing system is precise (the semantic mapping in an injection),
  - *valid* if each statement in the representing system is valid (the semantic mapping is verifiable),
  - *accurate* if it is both precise and valid (the semantic mapping is an injection and verifiable),
  - *incomplete* (underspecified) if some fact in the represented system does not correspond to a statement in the representing system (the semantic mapping is not a surjection),
  - *complete* if the semantic mapping is a bijection (injection and surjection), and
  - *partial* (over specified) if some statement in the representing system does not correspond to a fact in the represented system (the semantic mapping is a partial function).

The following observations are useful. A vague belief has a much better chance of being true than a precise one because there are more possible facts that would verify it. The more specific (precise) a claim, the less likely it is to be true. A precise belief is harder to be true but better worth having if it is true. "Science is trying to substitute more precise beliefs for vague ones; this makes it harder for a scientific proposition to be true than for vague beliefs but make scientific truth better worth having if it can be obtained." - B. Russell.

> **Aside:**Words have meaning by virtue of their correspondence to some action or object. Actions may have consequences. Objects and actions may have explanations in some context. A consequence

of a correspondence between words and objects in a structure is that words have meaning. Thus it is correct to say that the universe has no meaning.

Aside:A correspondence between a and b is

- decidable if given a and b, it is decidable that I(a)=b,
- computable, if given a, I(a) is computable,
- verifiable, if

7.4 Model Validation

A relational structure is a model of a domain of interest. In Shanks et. al [50], a model is a *faithful representation* of a domain if it posesses the following attributes.

- Accurate.
- Complete.
- Conflict-free.
- No redundancy.

Validation is the process of checking that the model faithfully represents the domain. Model validation is of particular interest in science and engineering. The process of validation must address three issues.

1. Scope.
2. The people involved in the process.
3. The methodology.

   There are several approaches:

   - Review.
   - Questioning.
   - Problem solving.
   - Transaction testing.

The following table contrasts various types of scholarship.

	Humanistic theory	**Scientific theory**	**Mathematical theory**
Universe of study	Discovering the person	A complex, dynamic universe.	An artificial, static, completely describable world.
Assumptions	Subjective world where the knower and the known cannot be separated.	Objective world where the knower and the known can be separated.	Objective world where the knower and the known can be separated.
Goal	Individuals, Give rules for interpretation, Time dependent	To discover those principles and relations that are invariant across, time, instances of application, transformations of coordinate systems and representational systems of the phenomena	
Items of interest	Non-deterministic phenomena	Deterministic phenomena	deterministic phenomena

By universe of interest

- Access: subjective; objective
- Description: Complete; incomplete
- Behavior:
  - Static;
  - dynamic/variation
    - deterministic;
    - nondeterministic;
      - stochastic; probabilistic
      - chaotic;
    - adaptive behavior

Idealized relational structures are constructed from empirical data.

7.5 Features of Empirical Data

Fuzzy Selection bias Fuzzy selection bias occurs when there is some degree of uncertainty as to whether the selected object satisfies some property.

Formally: Let U be the universe of data, $s \in U$, and P be some property for elements in U. If $P(s)$ could have some value between $[0, 1]$ then there is a fuzzy selection bias.

Confirmation bias The confirmation bias is the intuitive decision making process that sorts though a body of data and selects those that most confirm to what is already believed, and ignores or rationalizes away those that do not. This is in contrast to a formalized rational decision making process. The confirmation bias occurs in illusions stemming from ... ... counter-intuitive - limitations of informal human thought ... ... paradox ... ... coincidences and low probability events ...

Complete ...

Need to provide definition

Anthropic Bias

Selection bias Intuitively: The selection bias effect occurs when data samples are from a proper subset of the target domain rather than a representative random sample. These may occur due to limitations of the data collection device. Example: the size of the openings in a net determine the size of the fish collected which do may not be representative of the sizes that actually occur. Formally: Let U be the universe of data and let S be the set of data item selected. If $S \subset U$ (S is a proper subset of U) then there is selection bias with respect to some property P if for every $s \in S$, P(s) holds and there is a $q \in U \backslash S$ such that it is false that P(q). However, there is no selection bias if S is a random sample of U and S is of sufficient size to insure that it is similar to U with respect to the property P (the sampling density $|S|/|U|$ to achieve this is unspecified). Of course if S=U then there is no selection bias.

Observation selection bias An additional precondition to selection bias effects which requires that somebody be there to "have" the data. The key difference between selection and observation bias is whether the theory of observation selection effects is necessary to model the bias.

Self-sampling assumption One should reason as if one were a random sample from the set of all observer-moments in one's reference class. The reference class is the class of entities such that one should reason as if one were randomly selected from it.

Software Engineering	**Science**
Needs document	Idea for study
Requirements; tools and methodology	Experimental design; Data collection; domain specific tools and methodology
Design	??
Implementation	??

Data collection tools and methodologies

Ethnographic research

Main technique: participant observation

- living among the user community
- gathering data through involvement with the user community

Stages

- Entry: open-ended data gathering learning the language and basic rules
- Systematic data gathering: administer formal interviews related to the project focus.

Strategies

- Judgement Sampling/Key Informant Interview: select a small representative sample of informants for detailed interview.
- Random sampling: get a limited amount of information from a large number of informants.

Measured data

- Real world is a fuzzy place
- Data usually has noise (random errors)
- Data must be smoothed
- Data is point sampled from an analog domain
- Potential aliasing artifacts
- Data may be wrong
- Error in experiment

Textual data

- textual criticism - reconstruction of the most probable text from existent variant texts and other evidence.
- cultural background
- what was meant rather than what was said

Further details are beyond the scope of this document.

The absolute, relative, objective and subjective

> Questions about objectiveness depend upon the range of transformations under which something is invariant.
> - Robert Nozick in *Invariances: the structure of the objective world.*

The goal of objective scientific theories of phenomena is to discover principles and relations which are invariant across, time, instances of application, transformations of coordinate systems and representational systems of the phenomena.

> The word "objectivity" refers to the view that the truth of a thing is independent from the observing subject. The notion of objectivity entails that certain things exist independently from the mind, or that they are at least in an external sphere. Objective truths are independent of human wishes and beliefs. The notion of objectivity is especially relevant to the status of our various ideas, and the question is to what extent objectivity is possible for thought, and to what extent it is necessary.
>
> In epistemology, the objectivist position is that truth independent of the individual; this follows the correspondence theory of truth. However, idealists use *objectivity* to designate that existence in thought is the only kind of real existence. In metaphysics, Plato identifies objectivity as pertaining to the world of the forms. For Plato, the forms reside in a separate world, which is invisible to our sense, although obtainable through reason. Thus, Plato refers to real objects the "knowable forms" which include the objective truths of justice, beauty, truth, and love.
>
> Philosophers of the modern period concede the reality of the objective realm, although argue that it is unattainable. This is so of Locke's account of the a thing's substance, and Kant's view that our knowledge is restricted to the phenomenal realm, with no direct access to things in themselves.
>
> In this century, Richard Rorty distinguishes between two notions of "objectivity." One involves the correspondence with what is out there, and is supposedly discovered by an algorithm. This Rorty rejects as since we have no idea how to perform this task. His second notion of "objective" involves those considerations adopted by a consensus of rational discussants. This, he believes, is the most objectivity we can hope for. From: Objectivity in the internet encyclopedia of philosophy `www.utm.edu/research/iep/o/objectiv.htm`

	Objectivity	**Subjectivity**
Observations	Observations hold across multiple observers and/or multiple instances following a common protocol - experimental outcomes are repeatable and independent of the investigator - replications of a study should yield identical results.	Observations are unique to the individual observer and /or observation. Are used where the information is incomplete, inconsistent, and/or unique.
Method	What protocol one follows determines what one sees.	Who one is, determines what one sees.
Trust	Trust in a protocol.	Trust in individuals.

A formalization of absolute, objective, relative and subjective properties.

(a proposal)

Let **L** and **S** be sets and **F** a subset of the functions from **L** to **S** i.e., $\mathbf{F}\subseteq\mathbf{L}\rightarrow\mathbf{S}$, and let **P** and **Q** be properties (not necessarily distinct) defined on **L**, **S**, and **F** i.e., **P** and **Q** are subsets of $\mathbf{L}\times\mathbf{S}\times\mathbf{F}$. Think of each human intelligence as being an element of **F**.

Absolute **P** is an *absolute property* if it is decidable and holds for all functions in **F**.

> **Definition 7.1** *P is* absolute *for the set of validation functions $\mathcal{V}$ if for every $v_i, v_j \in \mathcal{V}$, $v_i \models P = v_j \models P$.*
>
> Example: Tautologies are formulas that are true for all valuation functions. Another way of saying it is that *absolute* properties are those properties that are invariant under all transformations.

Objective **P** is an *objective property* if it is decidable. This class of properties includes the mathematical theories of geometry and algebra as well as scientific theories based on repeatable methodologies. Another way of saying it is that *objective* properties are those properties that are invariant under some transformations.

> **Definition 7.2** *P is* objective *for a set of validation functions $\mathcal{V}_c$ if for every $v_i, v_j \in \mathcal{V}_c \subset \mathcal{V}$, $v_i \models P = v_j \models P$.*

Relative **P** is a *relative property* if it holds whenever **Q** holds but does not hold for all functions in **F**.

Definition 7.3 *P is* relative *for the set of validation functions $\mathcal{V}$ if for every $v_i, v_j \in \mathcal{V}$, $v_i \models P \neq v_j \models P$.*

Examples include those properties that are logical consequences of the non-logical axioms. So a property might be true in the theory of groups but not in the theory of fields. An other example is any cosmological event of the distant past or future explained in terms of contemporary scientific theory.

Subjective **P** is a *subjective property* if it holds and is undecidable for at most one function in **F**. Another way of saying it is that *subjective* properties are those properties for which there are no invariant transformations.

Definition 7.4 *P is* subjective *for a set of validation functions $\mathcal{V}_c$ if for every $v_i, v_j \in \mathcal{V}_c \subset \mathcal{V}$, $v_i \models P \neq v_j \models P$.*

The absolute is usually associated with that which is complete, unconditional, or has independent existence. To say that "everything is relative" is to acknowledge the difficulty of verifying absolute properties. The choice here is to associate absolute with decidability (a computation concept) and universality within a particular domain. The particular domain must be carefully defined otherwise it is subject to the paradoxes of set theory.

Objectivity is usually associated with the notion that which is independent of an individual observer. The choice here is to associate it with the concept of decidability. There is a strong overlap between objective and relative properties. This definition does not take into account the evolving nature and the degree of acceptance of theories.

Proposition 1: Absolute $\subset$ Objective. *Proof*: Immediate consequence of the definitions. Example: Tautologies $\subset$ Theorems

Proposition 2: Subjective $\subset$ Relative. *Proof*: Let **Q** be **P**. The proposition is immediate of this assumption and the definitions. Example: Unsupported belief $\subset$ The consequences of unsupported belief

Proposition 3: Absolute $\cap$ Relative $= \emptyset$. *Proof*: Immediate consequence of the definitions. Example: Tautologies $\cap$ nonLogical Axioms $= \emptyset$

Proposition 4: Subjective $\cap$ Objective $= \emptyset$. *Proof*: Immediate consequence of the definitions. Example: Unsupported belief $\cap$ supported belief $= \emptyset$

7.6 Reading list and references

Bostrom, Nick *Anthropic Bias: Observation Selection Effects in Science and Philosophy.* Routledge 2002.

Irvine, A. "Alfred Tarski", `plato.stanford.edu/entries/tarski-truth`, The Stanford Encyclopedia of Philosophy. Edward N. Zalta (ed.)

Littlejohn, Stephen W. (1999) *Theories of Human Communication 6th ed.* Wadsworth Publishing Company 1999.

Mahoney, Michael S. (2000) Software as Science - Science as Software in *Proceedings of the International Conference on History of Computing 2000 Mapping the History of Computing: Software Issues* Heinz Nisdorf MuseumsForum, Paderborn, Germany, 5-7 April 2000.

Margenau, Henry (1950) *The nature of physical reality* McGraw-Hill Book Company, Inc.

Quine and Ullian (1988). Chapter 18 in *Introductory Readings in the Philosophy of Science*. Prometheus Books.

Part II

Logics

Chapter 8

Constructive (Intuitionistic) Logic

Intuitionists conclude that *the meaning of a statement resides not in its truth conditions but in the means of proof or verification.* In classical logic, disjunctive formulas of the form $P \vee \neg P$ are provable without providing a proof of either P or of $\neg P$ and some formulas of the form $\exists x.F(x)$ (such as $\exists x.\forall y.[p(x) \rightarrow p(y)]$) is provable without providing a proof of $F(t)$ for some particular t. *Intuitionism* is a school of philosophy of mathematics that questions these tenets of classical logic. Intuitionism demands a *constructive* interpretation of the quantifiers: if $\exists x.F$ is true, then the value of x satisfying F should be effectively computable. Thus intuitionistic proofs contain more information than classical proofs. Hence intuitionistic logic can be used for program synthesis. In intuitionistic type theory, a proof of $\exists x.F$ constructs a function to compute x. However, proof search in intuitionistic logic is more difficult than in first-order classsical logic; there are no normal forms like conjunctive normal form or prenex form and Skolemization cannot, in general, be applied to intuitionistic formulas.

The following principles were formulated by Brouwer, Heyting, and Kolmogorov and are called the **BHK**-interpretation of constructive logic.

1. A proof of $A \wedge B$ is given by presenting a proof of A and a proof of B.
2. A proof of $A \vee B$ is given by presenting either a proof of A or a proof of B and indicating which proof it is.
3. A proof of $A \rightarrow B$ is a procedure which permits us to transform a proof of A into a proof of B.
4. The constant *false*, a contradiction, has no proof.
5. A proof of $\neg A$ is a procedure that transforms any hypothetical proof of A into a proof of a contradiction.

8.1 Syntax

The set of atomic formulas, $\mathcal{P}$, is defined by

$\mathcal{P} = \{P^i_j t_k ... t_{k+i-1} \mid t_l \in \mathcal{C},\ i, j, k, l \in \mathbb{N}\}$ with $\mathbf{f} \in \mathcal{P}$ where $\mathcal{C} = \{F^i_j t_k ... t_{k+i-1} \mid t_k \in \mathcal{C},\ i, j, k \in \mathbb{N}\}$ is a set of *terms*,
$\{P^0_j \mid j \in \mathbb{N}\}$ is a set of *propositional constants*, and
$\{F^0_j \mid j \in \mathbb{N}\}$ is a set of *individual constants*.

The set of formulas, $\mathcal{F}$, is defined by

$\mathcal{F} ::= \mathcal{P} \mid\ \rightarrow \mathcal{F}\mathcal{F} \mid \Box\mathcal{F} \mid \forall x.[\mathcal{F}]^x_t$

where $\mathcal{V} = \{x_i \mid i \in \mathbb{N}\}$ is a set of *individual variables*, $t \in \mathcal{C}$, $x \in \mathcal{V}$, and textual substitution, $[F]^t_x$, is a part of the meta language and designates the formula that results from replacing each occurrence of t with x.

Additional operators and infix notation:

$(A \rightarrow B) \equiv\ \rightarrow AB$
$\neg A \equiv (A \rightarrow \mathbf{f})$
$(A \vee B) \equiv (\neg A \rightarrow B)$
$(A \wedge B) \equiv \neg(A \rightarrow \neg B)$
$\mathbf{f} \equiv (A \wedge \neg A)$
$(A \leftrightarrow B) \equiv ((A \rightarrow B) \wedge (B \rightarrow A))$
$\Diamond A \equiv \neg\Box\neg A$
$\exists x.A \equiv \neg\forall x.\neg A$

8.2 Axioms

The following are logical axioms for intuitionistic logic:

1. →A→BA
2. →→A→BC→→AB→AC
3. $\rightarrow \wedge ABA$
4. $\rightarrow \wedge ABB$
5. $\rightarrow A \rightarrow B \wedge AB$

6. $\to A \vee AB$
7. $\to B \vee AB$
8. $\to\to AC \to\to BC \to \vee ABC$
9. $\to \mathbf{f}A$
10. $\to \forall x.A[A]^x_c$ where x
11. $\to [A]^x_c \exists x.A$ where x
12. $\to \forall x. \to AB \to A\forall x.B$ where x is not free in A.
13. $\to \forall x. \to BA \to \exists x.BA$ where x is not free in A.

The inference rules are

1. (*modus ponens*) from A and $\to AB$ infer B
2. (*generalization*) from A, if x is a variable, infer $\forall x.A$.

8.3 Sequent Calculus for Intuitionistic Logic

- A sequent calculus for intuitionistic logic. $A \Leftrightarrow B = A \Rightarrow B \wedge B \to A$

Sequent Axiom and Inference rules for Intuitionistic Logic

Axioms	$[U, X \vdash V, X]$	$[U, \text{false} \vdash U, A$
Rules	*Left Rules*	*Left Rules*
	[*true formulas* $\vdash$ *false formulas*]	[*true formulas* $\vdash$ *false formulas*]
Negation	$\frac{[U,\neg F\vdash V]}{[U,\neg F\vdash V,F]}$	$\frac{[U\vdash V,\neg F]}{[U,F\vdash V,\text{false}]}$
And	$\frac{[U,\alpha\vdash V]}{[U,\alpha_1,\alpha_2\vdash V]}$	$\frac{[U\vdash V,\alpha]}{[U\vdash V,\alpha_1],[U\vdash V,\alpha_2]}$
Or	$\frac{[U,\beta\vdash V]}{[U,\beta_1\vdash V],[U\vdash V,\beta_2]}$	$\frac{[U\vdash V,\beta]}{[U,\beta_1,\beta_2\vdash V]}$
$\to$	$\frac{[U,A\to B\vdash V]}{[U,B\vdash V],[U,A\to B\vdash V,A]}$	$\frac{[U\vdash V,A\to B]}{[U,A\vdash V,B]}$
$\forall x$	$\frac{[U,\forall x.A\vdash V]}{[U,A(c),\forall x.A\vdash V]}$	$\frac{[U\vdash V,\forall x.A]}{[U\vdash V,A(c)]}$ c is new
$\exists x$	$\frac{[U,\exists xA\vdash V]}{[U,A(c)\vdash V]}$ c is new	$\frac{[U,\vdash V,\exists x.A]}{[U\vdash V,A(c)]}$

Some rules of thumb:

- In intuitionistic deduction, avoid using the beta right rules before beta left rules.
- Use delta left and gamma right before delta right and gamma left.

An implementation for intuitionistic first-order logic is `code/intseq` available

8.4 Kripke Model (Possible Worlds)

Let $\mathcal{W}$ be a set of worlds (valuation functions) and A a reflexive, transitive accesibility relation on $\mathcal{W} \times \mathcal{W}$. A Kripke Model, $\mathcal{M} = \langle \mathcal{W}, \mathcal{A} \rangle$, is a graph whose edges are labeled with literal formulas (the formulas required to be true by the valuation function).

For all $a, b, c \in \mathcal{W}$

- A is reflexive and transitive.
- Aab implies $a \subset b$. Any sequence of constants is monotonic.
- Aab implies $a(At) \subset b(At)$. Any sequence of atomic formulas is monotonic.

For some Kripke model $\mathcal{M}$, $a \in \mathcal{W}$, and a formula A, $\mathcal{M} \models_a$ A is defined as follows:

Intuitionistic Logic

not $\mathcal{M} \models_a \mathbf{f}$	for all $a \in \mathcal{W}$
$\mathcal{M} \models_a p$	iff $p \in C_b$, all b such that Aab
$\mathcal{M} \models_a \neg p$	iff $p \notin C_a$
$\mathcal{M} \models_a \neg\neg F$	iff for some b such that Aab, $\mathcal{M} \models_b \neg F$
$\mathcal{M} \models_a \wedge AB$	iff $\mathcal{M} \models_a A$ and $\mathcal{M} \models_a B$
$\mathcal{M} \models_a \vee AB$	iff $\mathcal{M} \models_a A$ or $\mathcal{M} \models_a B$
$\mathcal{M} \models_a \rightarrow AB$	iff for all b such that Aab, $\mathcal{M} \models_b \neg A$ or $\mathcal{M} \models_b B$
$\mathcal{M} \models_a \neg \rightarrow AB$	iff for some b such that Aab, $\mathcal{M} \models_b A$ and $\mathcal{M} \models_b \neg B$
$\mathcal{M} \models_a \forall x.F$	iff for all b such that Aab, $\mathcal{M} \models_b [F]_x^c$ for all $c \in C$
$\mathcal{M} \models_a \neg\forall x.F$	iff for some b such that Aab, $\mathcal{M} \models_b [F]_x^c$ for some $c \in C$
$\mathcal{M} \models_a \exists x.F$	iff $\mathcal{M} \models_a [F]_x^c$ for some $c \in C$
$\mathcal{M} \models_a \neg\exists x.F$	iff $\mathcal{M} \models_a [F]_x^c$ for all $c \in C$

A formula A is *intuitionistically valid* iff $\mathcal{M} \models_a A$ for $\mathcal{M}$ and every a.
A formula F is said to be *valid* ($\models F$) iff it is valid in all models $\mathcal{M}$ ($\mathcal{M} \models F$ for all $\mathcal{M}$). A valid formula is called a *tautology*.

Exercises

1. Show that for some a, $\mathcal{M} \models_a \neg A$ iff for every b wuch that Aab, not $\mathcal{M} \models_b A$.
2. Show that if $\mathcal{M} \models_a A$ and Aab, then $\mathcal{M} \models_b A$.
3. Show that for some a, not $\mathcal{M} \models_a \vee A \neg A$.
4. Show that for some a, not $\mathcal{M} \models_a \rightarrow \neg\neg A$.
5. Show that for some a, not $\mathcal{M} \models_a \rightarrow \neg \forall x.A \exists x.\neg A$.

8.5 Natural Deduction

Natural deduction has an intuitionistic orientation.

Constructive Natural Deduction Rules

	Introduction Rules		**Elimination Rules**	
$\wedge$	$\frac{\vdash A, \vdash B}{A \wedge B}$		$\frac{\vdash A \wedge B}{A}$	$\frac{\vdash A \wedge B}{B}$
$\vee$	$\frac{\vdash A}{A \vee B}$	$\frac{\vdash B}{A \vee B}$	$\frac{\vdash A \vee B \text{ and } A \vdash C \text{ or } B \vdash C}{C}$	
$\rightarrow$	$\frac{A \vdash B}{A \rightarrow B}$		$\frac{\vdash A, \vdash A \rightarrow B}{B}$	
$\neg$			$\frac{\vdash \mathbf{f}}{B}$	
Contradiction	$\frac{\mathbf{f}}{B}$		Classical Rule:	$\frac{[\neg B]\mathbf{f}}{B}$
$\forall x.$	$\frac{P(x)}{\forall x.P(x)}$		$\frac{\forall x.P(x)}{[P(x)]^c_x}$	
$\exists x.$	$\frac{P(c)}{\exists x.[P(x)]^x_c}$			
Induction	$\frac{[A]^0_x, \rightarrow [A]^n_x [A]^{n+1}_x}{\forall x.A}$			

8.6 Constructive Type Theory

Type theory was originally developed with the aim of being a clarification of constructive mathematics. An introduction to type theory as a theory for program construction. Evaluation of a well typed program always terminates.

8.7 References

Nordstrom, Petersson, & Smith *Programming in Martin-Lof's Type Theory* Oxford University Press 1990.

Otten, Jens *ileanTAP: An Intuitionistic Theorem Prover* `http://aida.intellektik.informatik.th-darmstadt.de/~jeotten/ileanTAP/`

Chapter 9

Multi-valued Logics

Multi-valued logics have valuation functions that map atomic formulas to more than two values. Figure 1 provides several difference definitions of valuation functions.

$v : \mathcal{P} \rightarrow \{0, 1\}$	v is boolean valued as in classical logic
$v : \mathcal{P} \rightarrow [0, 1]$	v is infinite valued and is suitable for use in a probabilistic, fuzzy, and other continuous logics.
$v : \mathcal{P} \rightarrow \{\perp, 0, 1\}$	v identifies undefined formulas such as the truth of $y > 1/x$.
$v : \mathcal{P} \rightarrow \{0, ..., n\}$	v is a multivalued logic with a range of values such as a Likert scale for tests.

A valuation function v satisfies the following:

$$
\begin{array}{lcl}
v(\mathbf{f}) & = & 0 \\
v(p) & = & \left\{ \begin{array}{ll} 0 & \text{if } p \text{ is false} \\ 1 & \text{if } p \text{ is true} \end{array} \right. \quad \text{where } p \in \mathcal{P}.
\end{array}
$$

v may be extended to non atomic formulas as follows:

$$
\begin{array}{lcl}
v(\neg A) & = & 1 - v(A) \\
v(A \vee B) & = & \max(v(A), v(B)) \\
v(A \wedge B) & = & \min(v(A), v(B)) \\
v(A \rightarrow B) & = & \max(v(\neg A), v(B)) \\
v(\exists x.A) & = & \max_{c \in C}(v([A]_x^c) \\
v(\forall x.A) & = & \min_{c \in C}(v([A]_x^c)
\end{array}
$$

Valuation functions and the models relation in classical logic.

$$
v \models A \quad \text{iff} \quad v(A) = 1
$$

The challange for infinite valued logics if to find a way to determine thevalue of a proposition.

9.1 Syntax and Semantics

The syntax of the language L is described using the symbols of the language L, standard mathematical notation, and meta symbols from natural languages.

The set of atomic formulas, $\mathcal{P}$, is defined by

$\mathcal{P} = \{P^i_j t_k...t_{k+i-1} \mid t_l \in \mathcal{C},\ i,j,k,l \in \mathbb{N}\}$ with $\mathbf{f} \in \mathcal{P}$
where $\mathcal{C} = \{F^i_j t_k...t_{k+i-1} \mid t_k \in \mathcal{C},\ i,j,k \in \mathbb{N}\}$ is a set of *terms*,
$\{P^0_j \mid j \in \mathbb{N}\}$ is a set of *propositional constants*, and
$\{F^0_j \mid j \in \mathbb{N}\}$ is a set of *individual constants*.

The set of formulas, $\mathcal{F}$, is defined by

$\mathcal{F} ::= \mathcal{P} \mid\ \rightarrow \mathcal{F}\mathcal{F} \mid \Box\mathcal{F} \mid \forall x.[\mathcal{F}]^x_t$

where $\mathcal{V} = \{x_i \mid i \in \mathbb{N}\}$ is a set of *individual variables*, $t \in \mathcal{C}$, $x \in \mathcal{V}$, and textual substitution, $[F]^t_x$, is a part of the meta language and designates the formula that results from replacing each occurrence of t with x.

Additional operators and infix notation:

$(A \rightarrow B) \equiv\rightarrow AB$
$\neg A \equiv (A \rightarrow \mathbf{f})$
$(A \vee B) \equiv (\neg A \rightarrow B)$
$(A \wedge B) \equiv \neg(A \rightarrow \neg B)$
$\mathbf{f} \equiv (A \wedge \neg A)$
$(A \leftrightarrow B) \equiv ((A \rightarrow B) \wedge (B \rightarrow A))$
$\Diamond A \equiv \neg\Box\neg A$
$\exists x.A \equiv \neg\forall x.\neg A$

There is a *valuation* function v from formulas to the interval [0,1]. The function v is a total function on the set of atomic formulas. It is assumed that the valuation functions must be *generalizations of the valuation functions for classical logic.* Figure N.1 summarizes these concepts.

Some definitions of suitable valuation functions are given in the table below. Some of these definitions do not preserve deMorgan's laws.

$v(\neg A) = 1 - v(A)$
$v(A \vee B) = max(v(A), v(B))$
$v(A \vee B) = min(1, v(A) + v(B))$
$v(A \vee B) = v(A) + v(B) - v(A) \times v(B)$
$v(A \wedge B) = min(v(A), v(B))$
$v(A \wedge B) = max(0, v(A) + v(B) - 1)$
$v(A \wedge B) = v(A) \times v(B)$
$v(A \rightarrow B) = max(1 - v(A), v(B))$
$v(A \rightarrow B) = min(1, 1 - v(A) + v(B))$
$v(A \rightarrow B) = 1 - v(A) + v(A) \times v(B)$
$v(A \leftrightarrow B) = 1 - |v(A) - v(B)|$
$v(A \leftrightarrow B) = min(max(1 - v(A), v(B)), max(1 - v(B), v(A)))$
$v(A \leftrightarrow B) = max(min(v(A), v(B)), min(1 - v(A), 1 - v(B)))$
$v(A \leftrightarrow B) = max(0, min(1, 1 - v(A) + v(B)) + min(1, 1 - v(B) + v(A)) - 1)$
$v(A \leftrightarrow B) =$ other based on various properties

9.2 Inference Rules

Alpha	$\frac{A \wedge B}{A \mid B}$	Both subformulas hold.
Beta	$\frac{A \vee B}{A \mid B}$	Branch since only one of the subformulas may hold
Gamma	$\frac{\forall x.A}{[A]_x^c, \forall x.A}$	If A holds for all x then $[A]_x^c$ holds for any constant c
Delta	$\frac{\exists x.A}{[A]_x^c}$	If there exists an x such that A holds, then conclude $[A]_x^c$ holds for a *new* constant
Modus Ponens	$\frac{A, A \rightarrow B}{B}$	If A holds and the rule $A \rightarrow B$ holds, then B holds.
Modus Tollens	$\frac{\neg B, A \rightarrow B}{\neg A}$	If B is false and the rule $A \rightarrow B$ holds, then A is false.
Default	$\frac{A : B, \ldots}{C}$	If A holds and the B are consistent, then C holds.

Lukasiewicz valuation function for multi-valued logics

Exercises

1. Show how to derive the various valuation functions using deMorgan's rules.

9.3 Reasoning with multi-valued logic

Inference rules

- *Modus ponens*: If A and $A \rightarrow B$ are valid then so is B i.e.,

$A[x], (A[x] \to B[y])[z] | B[y]$ so $y = f(x, z)$
$y \geq x$ if $0.5 \leq x \leq z$.

- *Modus tollens*: If $\neg B$ and $A \to B$ are valid, then so is $\neg A$. i.e.,
  $\neg B[y], (A[x] \to B[y])[z] | \neg A[x]$ so $x = f(y, z)$.

A small implementation is ¡a href=”code/mvl”¿available.

9.4 Fuzzy Logic

Fuzzy Predicates: tall, young, small, medium , normal, expensive near, intelligent, ...
Fuzzy Truth values: true false, fairly true, very true
Fuzzy probabilities: likely, unlikely, very likely highly unlikely
Fuzzy quantifiers: many few, most, almost all

9.5 References

Stanford Encyclopedia of Philosophy `http://plato.stanford.edu/entries/logic-manyvalued` Many-valued Logic

Continuous Logic

Poli, R., Ryan, M., Sloman, A *A New Continous Propositional Logic* Technical Report: CSRP-95-9; University of Birmingham 1995

Default logic

Besnard, Philippe *An Introduction to Default Logic* Springer-Verlag 1989

Fuzzy logic

Klir, St. Clair, & Yuan (1997) *Fuzzy Set Theory: Foundations and Applications* Prentice-Hall 1997
Klir & Yuan (1995) *Fuzzy Sets and Fuzzy Logic: Theory and Applications* Prentice-Hall 1995 ¡/dl¿

Implementation

Sterling and Shapiro (1986) *The Art of Prolog* MIT Press 1986

Chapter 10

Non-monotonic Logics

Traditional logics are based on *sound deduction*, a method of exact inference with the advantage that its conclusions are exact - there is no possibility of mistake if the rules are followed exactly. *Deduction requires that information be complete, precise, and consistent.* By contrast, the real world requires common sense reasoning in the face of incomplete, inexact, and potentially inconsistent information.

A logic is *monotonic* if the truth of a proposition does not change when new information (axioms) are added to the system. In contrast, a logic is *non-monotonic* if *the truth of a proposition may change* when new information (axioms) is added to or old information is deleted from the system.

Inference Rules:

Definition 10.1 Abduction*, infer plausible causes of an effect. The abduction inference rule is:*

$$\frac{Q(c),\ P(a) \Rightarrow Q(a)}{P(c)}$$

where $\Rightarrow$ is a causal implication and the conclusion is plausible rather than necessary thus differing from modus tollens.

Definition 10.2 Deduction*, exact inference. Modus ponens and specialization are the two primary inference rules of deduction.*

$$\frac{P,\ P \to Q}{Q},\ \frac{\forall x.P(x)}{P(c)}$$

Definition 10.3 Induction*, infer generalizations from a set of events. The induction inference rule is:*

$$\frac{P(c)}{\forall x.P(x)}$$

Definition 10.4 Default logic*: Inference rule*

$$\frac{P : D}{C}$$

If p *is true but* D *is unknown, then* C*.*

10.1 Default Logic

The language of default logic is the language of first-order logic. The rules of inference in default logic are called *defaults* and are expressions of the form:

$$\frac{\alpha(x) : \beta_1(x), ..., \beta_m(x)}{\gamma(x)}$$

where $m \geq 0$ and $\alpha(x)$, the $\beta_i(x)$, and $\gamma(x)$ are formulas of first-order logic. The formula $\alpha(x)$ is called the *prerequisite* of the default rule, the $\beta_i(x)$s are called the *justifications*, and the formula $\gamma(x)$ is called the *conclusion*. The intuitive meaning of a default is that for every n-tuple $\mathbf{t}$ of terms, if $\alpha(\mathbf{t})$ is believed and each of the $\beta_i(\mathbf{t})$s is consistent with one's beliefs, then one is permitted to deduce $\gamma(\mathbf{t})$ and add it to one's belief set.

A *default theory* is is a pair $\langle D, A\rangle$ where D is a set of defaults, and A is a set of axioms.

10.2 Autoepistemic Logic

The language of first-order autoepistemic logic is the language of first-order modal logic which is an extension of first-order logic with the unary modal connective **B** (read as believed).

Theorem 10.1 *Autoepistemic logic can be embedded in default logic.*

10.3 References

¡dl¿ ¡dt¿Chesevar, Carlos I. and Maguitman, Ana G.¡/dt¿ ¡dd¿Logical Models of Argument in ACM Computing Surveys. Vol 32. Number 4. December 2000 pp. 337-383.¡/dd¿ ¡dt¿Kaminski, M. and Rey, G. 2002.¡/dt¿ ¡dd¿Revisiting Quantification in Autoepistemic Logic ¡em¿ACM Transactions on Computational Logic¡/em¿, Vol. 3, No. 4, October 2002. pp. 542-561.¡/dd¿ ¡dt¿Stanford Encyclopedia of Philosophy¡/dt¿ ¡dd¿¡a href="http://plato.stanford.edu/entries/logic-nonmonotonic"¿Non-Monotonic Logic¡/a¿¡/dd¿ ¡/dl¿

Chapter 11

Modal Logics

Modal logics are designed to express possibility, necessity, belief, knowledge, temporal progression and other modalities. It is customary to add the operator $\Box$ with the interpretation determined by the logic. A second operator $\Diamond$ is the dual of the first i.e. $\Diamond A = \neg\Box\neg A$ and $\Box A = \neg\Diamond\neg A$. Figure 1 illustrates some readings of the formulas $\Box A$ and $\Diamond A$.

Reading of $\Box A$ and $\Diamond A$

	$\Box A$		$\Diamond A$
Necessity	A is necessary.	Possibly	A is possible.
Belief	A is believed.		
Knowledge	A is known.		
Time	A is always true.	Eventually	A is eventually true.

There are modal logics that can be used to express ideas such as:

- It might rain tonight.
- Life is unfair.
- Mary believes that John loves her.
- I know that you know that I know that you know I will be leaving town tomorrow.
- He went to town for some supplies, is now carving a duck, and when he is finished, he will paint it.

The concept of necessity can be understood in several different contexts:

- *Logical necessity*: logic requires it to be so. if A and $A \to B$ are true , then B must be true.

- *Epistemic necessity*: reality requires it to be so. What goes up must come down.
- *Moral necessity*: morality requires it to be so. Sinners will be punished.
- *Temporal necessity*: Since Camile was born in 1985, she must be at least 14 years old.

Propositional modal logics provide some of the expressive power of both first and second order logic and find applications in

- Artificial intelligence research area such as
  - natural language translation and
  - reasoning systems dealing with theories of knowledge, belief, and time.
- Database systems
- Software engineering
  - Program specification
  - Program verification
  - Protocol specification
- Theories of program behavior
  - Algorithmic logic
  - dynamic logic
  - process logic
  - temporal logic

Temporal logics are designed to express temporal progression. Temporal logic plays an important role in the specification, derivation, and verification of programs as programs may be viewed as progressing through a sequence of states, a new state after each event in the system. They have a particularly useful role in the specification and verification of communication protocols and reactive systems.

The key notion in the semantics of modal logic is the notion of possible worlds.

11.1 Syntax

For simplicity and ease of presentation, only propositional logics are used.

Let $\mathcal{P} = \{p_i | i = 0, 1, ...\}$ be a set of *atomic formulas* and $P \in \mathcal{P}$
Formulas

$\mathcal{U} = \langle \mathcal{W}, \mathcal{A} \rangle$ – a multiworld universe

$\mathcal{W} = \{R \mid R$ is a relational structure$\}$ – a set of worlds.
$\mathcal{A} \subset \mathcal{W} \times \mathcal{W}$ – an accessability relation between worlds.

Figure 11.1: Multiple world structure $\mathcal{U} = \langle W, A \rangle$

$F ::= P \mid \neg F \mid (F \vee F) \mid (F \wedge F) \mid (F \rightarrow F) \mid (F \leftrightarrow F)$
$F ::= P \mid \neg F \mid \wedge FF$ *The set of propositional formulas.*
$F ::= P \mid \mathbf{f} \mid \rightarrow FF$ *The set of propositional formulas.*
$F ::= P \mid \mathbf{f} \mid \rightarrow FF \mid \Box F$ *The set of propositional formulas.*
$F ::= P \mid \mathbf{f} \mid \rightarrow FF \mid \mathcal{B}F$ *The set of propositional formulas.*
$F ::= P \mid \mathbf{f} \mid \rightarrow FF \mid \mathcal{K}_i F \mid \mathcal{E}_G F \mid \mathcal{C}_G F$ *The set of propositional formulas.*

The prefix syntax was chosen for two reasons. First to be minimal and second to avoid the use of parentheses. Informally, we use the more common infix notation with parentheses and introduce additional logical operators via syntactic definitions.

Constants: $C = \{f^0_j \mid; j \in \mathbb{N}\}$
Functions: $f^i_j(t_1, t_2, ..., t_i)$ for $f^i_j t_1 t_2 ... t_i$ and $i > 0$
Terms: the constants and functions
Predicates: $p^i_j(t_1, t_2, ..., t_i)$ for $p^i_j t_1 t_2 ... t_i$
Negation: $\neg A = \rightarrow A\mathbf{f}$
Disjunction: $(A \vee B) = \rightarrow \neg AB$
Conjunction: $(A \wedge B) = \neg \rightarrow A \neg B$
Conditional: $(A \rightarrow B) = (\neg A \vee B$
Biconditional: $(A \leftrightarrow B) = ((A \rightarrow B) \wedge (B \rightarrow A))$
Existential Quantifier: $\exists x.A = \neg \forall x \neg A$
Diamond: $\Diamond A = \neg \Box \neg A$

The axioms are the tautologies and the inference rule is *modus ponens.*

11.2 Semantics - Multiple Worlds (Saul Kripke)

Multiple structures (see Figure 11.1) can be used to model *modal* formulas.

The set of constants in each world is monotonic in the accessibility relation however, the worlds may differ in the atomic formulas that hold for each world. $\mathcal{U}$ is a graph whose edges are labeled with literal formulas (the formulas required

$\mathcal{M} = \langle W, A \rangle$ is a graph where
$W = \{w \mid w$ is a relational structure$\}$
$A \subset W \times W$ – is the accessibility relation

$M \models \bot$	iff $v(\bot) =$ false
$M \models \top$	iff $v(\top) =$ true
$M \models p$	iff $v(p) \in w$
$M \models \neg A$	iff not $M \models A$
$M \models (A \vee B)$	iff $M \models A$ or $M \models B$
$M \models (A \wedge B)$	iff $M \models A$ and $M \models B$
$M \models A \rightarrow B$	iff $M \models \neg A$ or $M \models B$
$M \models \neg(A \rightarrow B)$	iff $M \models A$ or $M \models \neg B$
$M \models \Box A$	iff $M' \models A$ for all u such that Awu and $M' = (U, u, v)$
$M \models \Diamond A$	iff $M' \models A$ for some u such that Awu and $M' = (U, u, v)$
$M \models \forall x F$	iff $M \models [F]^c_x$ for all $c \in C$
$M \models \neg \forall x F$	iff $M \models \neg [F]^c_x$ for some $c \in C$

Figure 11.2: Modal Logic Semantics

to be true by the valuation function). A model $\mathcal{M} = \langle W, A, w, v \rangle$ is a Kripke structure. A Kripke structure where $|W| = 1$ corresponds to traditional logics. A reachability relation that is symmetric (Axy implies Ayx) implies that the graph is nondirectional. A reachability relation that is transitive (Axy and Ayz implie Axz) can be used to model temporal phenomenon. A reachability relation that is reflexive (Axx), symmetric, and transitive, can be used to reason about finite state systems.

The propositional modal logics share with classical propositional logic the finite model property; if a collection of formulas is satisfiable, it is satisfiable in a finite graph.

There are many modal logics. Table 11.2 illustrates the approach to semantics for modal logics.

A formula F is *valid* (a *tautology*), $\models F$, iff for all w in W, $\mathcal{M} \models F$ i.e., F *is true in all possible worlds.*

Linear time temporal logic is an example of a logic that uses multiple world semantics. Each time increment is represented by a world. The accessibility relation is reflexive and transitive but not symmetric as we assume that time does not run backwards. For the formula $\Box A$, A holds in the current world and in all future worlds and for the formula $\Diamond A$, A holds in either the current world or some future world.

Program specifications in temporal logic:

- Safety properties: $\Box P$
- Liveness properties: $\Diamond P$
- Safe-livenes property: $\Box(A \rightarrow \Diamond B)$
- The end of time: $\neg\Box\Diamond A$

Additional temporal operators include

- $\bigcirc P$ - next time
- $P\mathcal{U}Q$ - P until Q

The some of the temporal operators have recursive definitions:

- $\Box A = A \wedge \bigcirc\Box A$
- $\Diamond A = A \vee \bigcirc\Diamond A$
- $A\mathcal{U}B = B \vee (A \wedge \bigcirc(A\mathcal{U}B)$

Definition 11.1

- *A sentence S of L is* valid*, $\models S$, if it is true in all structures for L.*
- *A sentence S of L is a* logical consequence *of a set of sentences Ss of L ($Ss \models S$), if S is true in every structure in which all of the members of Ss are true.*
- *A set of sentences Ss, is* satisfiable *if there is a structure A in which all of the members of Ss are true. Such a structure is called a* model *of Ss. If Ss has no model, it is* unsatisfiable.

Proofs in classical logic concern truth in a single state while proofs in modal logics may involve several states. Since a formula may refer to a state other than the one in which it appears, once the collection of states has been constructed, the states must be checked to determine that all such references are satisfied.

¡a href=”ModTab.html”¿Tableau rules¡/a¿ for modal logic are available. ¡br¿ An implementation for ¡a href=”code/pml”¿propositional modal logic is available¡/a¿. ¡br¿ An implementation for ¡a href=”code/ml”¿first-order modal logic is available¡/a¿.

Proof Theory

In classical logic, the idea was to systematically search for a structure agreeing with the starting sentences. The result being that we get such a structure or each possible analysis leads to a contradiction. In modal logic, we try to build a frame agreeing with the sentences or see that all attempts lead to contradictions.

11.3 The Accessibility Relation

Gore 1992 has a wonderful list of axioms, a naming scheme

Property	**Axiom**	**Tableau rule**
reflexive	**T:** $\Box A \rightarrow A$	
symmetric	**B:** $A \rightarrow \Box\Diamond A$	
transitive	**4:** $\Box A \rightarrow \Box\Box A$	
serial	**D:** $\Box A \rightarrow \Diamond A$	

11.4 Models

K(ripke) - minimal modal logic

Axioms

1. All propositional tautologies
2. $\Box(p \rightarrow q) \rightarrow (\Box p \rightarrow \Box q)$

Rules of inference

1. from p and $p \rightarrow q$, infer q.

A theorem prover for ¡a href=”code/K”¿K is available¡/a¿.

B - self knowledge and knowledge of falsehoods

reflexive and symmetric
Axioms

1. All propositional tautologies
2. $\Box p \rightarrow p$ (reflexivity)
3. $\Box(p \rightarrow q) \rightarrow (\Box p \rightarrow \Box q)$
4. $\Box p \rightarrow \Box\Box p$ (transitivity)

The accessibility relation must be reflexive and transitive.
Rules of inference

1. from p and $p \rightarrow q$, infer q.
2. from p, infer $\Box p$.

D, D4, K, K4, K5, KB

T - knowledge is true belief

reflexive

M

Axioms

1. All propositional tautologies
2. $\Box p \rightarrow p$ (reflexivity)
3. $\Box(p \rightarrow q) \rightarrow (\Box p \rightarrow \Box q)$

Rules of inference

1. from p and $p \rightarrow q$, infer q.
2. from p, infer $\Box p$.

S4 - positive self reflection

reflexive and transitive
Axioms

1. All propositional tautologies
2. $\Box p \rightarrow p$ (reflexivity)
3. $\Box(p \rightarrow q) \rightarrow (\Box p \rightarrow \Box q)$
4. $\Box p \rightarrow \Box\Box p$

Rules of inference

1. from p and $p \rightarrow q$, infer q.
2. from p, infer $\Box p$.

S5, E - knowledge of non-knowledge (complete awareness)

reflexive, symmetric, transitive
Axioms

1. All propositional tautologies

2. $\Box p \rightarrow p$
3. $\Box(p \rightarrow q) \rightarrow (\Box p \rightarrow \Box q)$
4. $\Box p \rightarrow \Box\Box p$
5. $\Diamond p \rightarrow \Box\Diamond p$

Rules of Inference

1. from p and $p \rightarrow q$, infer q
2. from p, infer $\Box p$

11.5 Exercises

1. Show that $\Box A \equiv \neg\Diamond\neg A$.

11.6 References

Gore, Rajeev Prabharkar. *Cut-free Sequent and Tableau Systems for Propositional Normal Modal Logics.* (1992)

Stanford Encylopedia of Philosophy "Modal Logic" `http://plato.stanford.edu/entries/logic-modal`

Beckert, Bernhard and Gore, Rajeev "ModLeanTAP" `http://i12www.ira.uka.de/modlean`

Advances in Modal Logic `http://turing.wins.uva.nl/~mdr/AiML/`

Chapter 12

Belief, Knowledge and Self-Awareness

What kind of logic allows us to reason about our set of beliefs and our own self-awareness?

The following is adapted from Raymond Smullyan's book *Forever Undecided*.

Reasoning about beliefs requires a set of beliefs and a logic. The following is a description of a propositional belief logic based on propositional logic extended with the symbol $\mathcal{B}$; $\mathcal{B}X$ is read 'believes X' with the meaning that X is in the set of beliefs (See Figure N.1 and N.2).

Let $\mathcal{P} = \{p_i | i = 0, 1, ...\}$ be a set of *atomic formulas* and $P \in \mathcal{P}$
Formulas

$F ::= P \mid \neg F \mid (F \vee F) \mid (F \wedge F) \mid (F \rightarrow F) \mid (F \leftrightarrow F)$ $F ::= P \mid \neg F \mid \wedge FF$ *The set of propositional formulas.*
$F ::= P \mid \mathbf{f} \mid \rightarrow FF$ *The set of propositional formulas.*
$F ::= P \mid \mathbf{f} \mid \rightarrow FF \mid \Box F$ *The set of propositional formulas.*
$F ::= P \mid \mathbf{f} \mid \rightarrow FF \mid \mathcal{B}F$ *The set of propositional formulas.*
$F ::= P \mid \mathbf{f} \mid \rightarrow FF \mid \mathcal{K}_i F \mid \mathcal{E}_G F \mid \mathcal{C}_G F$ *The set of propositional formulas.*

The prefix syntax was chosen for two reasons. First to be minimal and second to avoid the use of parentheses. Informally, we use the more common infix notation with parentheses and introduce additional logical operators via syntactic definitions.

Constants: $C = \{f_j^0 \mid ; j \in \mathbb{N}\}$
Functions: $f_j^i(t_1, t_2, ..., t_i)$ for $f_j^i t_1 t_2 ... t_i$ and $i > 0$
Terms: the constants and functions

Predicates: $p^i_j(t_1, t_2, ..., t_i)$ for $p^i_j t_1 t_2 ... t_i$
Negation: $\neg A = \rightarrow A\mathbf{f}$
Disjunction: $(A \vee B) = \rightarrow \neg AB$
Conjunction: $(A \wedge B) = \neg \rightarrow A \neg B$
Conditional: $(A \rightarrow B) = (\neg A \vee B$
Biconditional: $(A \leftrightarrow B) = ((A \rightarrow B) \wedge (B \rightarrow A))$
Existential Quantifier: $\exists x.A = \neg \forall x \neg A$
Diamond: $\Diamond A = \neg \Box \neg A$

The semantics of Belief Logic include a set of beliefs and extend the valuation function of propositional logic to formulas containing the belief operator. Since beliefs are collections of formulas, we map a formula to true iff it is in the list of beliefs.

$\mathcal{M} = \langle SB, v \rangle$ where
SB is a set of formulas called beliefs}
$v : F \rightarrow \{0, 1\}$ – is a valuation function

not $M \models \mathbf{f}$	
$M \models P$	iff $v(p) = 1$
$M \models \rightarrow AB$	iff $M \models B$ or not $M \models A$
$M \models \mathcal{B}A$	iff $SB \models A$

A formula A is *satisfiable* iff it is true under some valuation function v and a set of beliefs SB, i.e., $\langle SB, v \rangle \models A$. A formula A is a *tautology* iff it is true under all valuation functions. A tautology is said to be *valid* and is written $\models A$. The axioms and rules of inference are:

Axioms	All tautologies
Rule of inference:	*Modus Ponens*

A set of formulas is said to be *logically closed* iff it *contains all tautologies* and is *closed under modus ponens* (for any formulas A and B, if A and $\rightarrow AB$ are in the set, then so is B).

A logically closed set of formulas is said to be *inconsistent* iff it *contains both a formula and its negation*, i.e., there is a formula A in the set such that both A and $\neg A$ are in the set. Equivalently, a set of formulas is said to be *inconsistent* iff it *contains all formulas*. A logically closed set of formulas is said to be *consistent* if it is not inconsistent.

Theorem 12.1 *The two definitions of inconsistent are equivalent.*

Proof: Clearly, the second definition implies the first. So, let F be a logically closed set of formulas that contains A and $\neg A$. Let Q

be an arbitrary formula. The formulas, $A \rightarrow \neg A \rightarrow A \wedge \neg A$, and $A \wedge \neg A \rightarrow Q$, are tautologies and so are in F. By three applications of MP, Q is in F.

Definitions

Definition 12.1 *A reasoner is called* accurate *if for any proposition A, if (s)he believes A, then A is true;*

$$\mathcal{B}A \rightarrow A$$

A reasoner is called inaccurate *if for some proposition A, if (s)he believes A, then ¬A is true.*

$$\mathcal{B}A \rightarrow \neg A$$

A reasoner is called consistent *if the set of all propositions the (s)he believes is a consistent set.*

$$\neg(\mathcal{B}A \wedge \mathcal{B}\neg A)$$

A reasoner is called normal *if for any proposition P, if (s)he believes A, then (s)he believes that (s)he believes A.*

$$\mathcal{B}A \rightarrow \mathcal{B}\mathcal{B}A$$

A reasoner is called peculiar *if for some proposition A, (s)he believes A and (s)he believes that (s)he doesn't believe A.*

$$\mathcal{B}A \wedge \mathcal{B}\neg\mathcal{B}A)$$

A reasoner is called regular *if for any propositions A and B, if (s)he believes $A \rightarrow B$, then (s)he also believes $\mathcal{B}A \rightarrow \mathcal{B}B$.*

$$\mathcal{B}(A \rightarrow B) \rightarrow \mathcal{B}(\mathcal{B}A \rightarrow \mathcal{B}B)$$

Observations: A reasoner *believes* that (s)he is consistent if for all formulas, A, (s)he $\mathcal{B}\neg\mathcal{B}(A \wedge \neg A)$. A reasoner *believes* that (s)he is inconsistent if for some formula A, (s)he believes $\mathcal{B}\mathcal{B}(A \wedge \neg A)$.

12.1 Advancing stages of self-awareness

What does it mean for an individual to be self-aware? People are aware of some external and internal events (their thoughts) and are able to recognize their

image. They are not aware of their atoms. We interpret the operator $\mathcal{B}$ to mean *aware of*.

Formula	Means
$\mathcal{B}A$	I am aware of A
$\mathcal{B}\mathcal{B}A$	I am aware of my belief of A
$\mathcal{B}\mathcal{B}\mathcal{B}A$	I am aware that I am aware of my belief of A

Now we define several types of reasoners.

1. A reasoner is of **type 0** if the set of beliefs are the tautologies. (S)he believes all tautologies, i.e., if $\models X$ (X is a tautology), then $\langle v, SB\rangle \models \mathcal{B}X$.

2. A reasoner is of **type 1** if the reasoner is of type 0 and the set of the reasoner's beliefs are logically closed i.e., If (s)he believes A and believes $A \to B$ then (s)he believes B i.e.,

$$\frac{\text{From: } \langle v, SB\rangle \models \mathcal{B}A \text{ and } \langle v, SB\rangle \models \mathcal{B}(A \to B)}{\text{infer: } \langle v, SB\rangle \models \mathcal{B}B}$$

3. A reasoner is of **type 2** if the reasoner is of type 1 and believes that her/his set of beliefs is logically closed i.e.,

$$\langle v, SB\rangle \models \mathcal{B}((\mathcal{B}A \land \mathcal{B}(A \to B)) \to \mathcal{B}B$$

for any formulas A and B. Reasoners of type 2 know how they reason – know their inference rule.

4. A reasoner is of **type 3** if the reasoner is of type 2 and is *aware of her/his beliefs* i.e.,

$$\langle v, SB\rangle \models \mathcal{B}A \to \mathcal{B}\mathcal{B}A$$

for any formula A. Any reasoner that is aware of her/his beliefs is said to be *normal*.

5. A reasoner is of **type 4** if the reasoner is of type 3 and is *aware that (s)he is aware of her/his beliefs* i.e.,

$$\langle v, SB\rangle \models \mathcal{B}(\mathcal{B}A \to \mathcal{B}\mathcal{B}A)$$

for any proposition A. A reasoner of type 4 *knows* that (s)he is normal.

Exercises/Theorems

1. Explain: Reasoners of type 2 *know* how they reason – they know their inference rule.

2. Explain: A reasoner of type 4 *knows* that (s)he is normal.

3. Prove that every reasoner of type 3 believes the proposition: $(\mathcal{B}p \wedge \mathcal{B}\neg p) \rightarrow \mathcal{B}(p \wedge \neg p)$.

   Proof: The following hold for type 3 reasoners:
   $\mathcal{B}A \rightarrow \mathcal{B}\mathcal{B}A$ and
   $\mathcal{B}A \rightarrow \mathcal{B}\mathcal{B}A$.
   $\mathcal{B}((\mathcal{B}A \wedge \mathcal{B}(A \rightarrow B)) \rightarrow \mathcal{B}B)$.
   Assume $\neg\mathcal{B}[(\mathcal{B}p \wedge \mathcal{B}\neg p) \rightarrow \mathcal{B}(p \wedge \neg p)]$ for some p.
   $\mathcal{B}p \wedge \mathcal{B}\neg p$, $\neg\mathcal{B}(p \wedge \neg p)$

- Prove that every reasoner of type 3 is regular.
- Prove that if a regular reasoner of type 1 believes ¡b¿¡i¿B¡/i¿¡/b¿A for some proposition A then (s)he must be normal.
- Prove that any peculiar normal reasoner of type 1 must be inconsistent.
- Prove that every reasoner of type 4 knows that (s)he is normal.
- Prove that any reasoner of type 4 knows that if (s)he should ever be peculiar, (s)he will be inconsistent.

12.2 Awareness of Self-Awareness

A reasoner believes that (s)he is of type 1 if (s)he believes all propositions of the form: $\mathcal{B}X$ where X is a tautology and believes all propositions of the form: $(\mathcal{B}A \wedge \mathcal{B}(A \rightarrow B)) \rightarrow \mathcal{B}B$.

$$\mathcal{B}\mathcal{B}X - X \text{ is a tautology}$$

$$\mathcal{B}((\mathcal{B}A \wedge \mathcal{B}(A \rightarrow B)) \rightarrow \mathcal{B}B)$$

A reasoner believes that (s)he is of type 2 if (s)he believes that (s)he is of type 1 and believes all propositions of the form: $\mathcal{B}((\mathcal{B}A \wedge \mathcal{B}(A \rightarrow B)) \rightarrow \mathcal{B}B$.

$$\mathcal{B}\mathcal{B}X - X \text{ is a tautology}$$

$$\mathcal{B}((\mathcal{B}A \wedge \mathcal{B}(A \rightarrow B)) \rightarrow \mathcal{B}B)$$

$$\mathcal{B}(\mathcal{B}((\mathcal{B}A \wedge \mathcal{B}(A \rightarrow B)) \rightarrow \mathcal{B}B)$$

A reasoner believes that (s)he is of type 3 if (s)he believes that (s)he is of type 2 and believes all propositions of the form: $(\mathcal{B}A \to \mathcal{B}\mathcal{B}A)$.

$$\mathcal{B}\mathcal{B}X - X \text{ is a tautology}$$
$$\mathcal{B}((\mathcal{B}A \wedge \mathcal{B}(A \to B)) \to \mathcal{B}B)$$
$$\mathcal{B}(\mathcal{B}((\mathcal{B}A \wedge \mathcal{B}(A \to B)) \to \mathcal{B}B)$$
$$\mathcal{B}(\mathcal{B}A \to \mathcal{B}\mathcal{B}A)$$

A reasoner believes that (s)he is of type 4 if (s)he believes that (s)he is of type 3 and believes all propositions of the form: $\mathcal{B}(\mathcal{B}A \to \mathcal{B}\mathcal{B}A)$.

$$\mathcal{B}\mathcal{B}X - X \text{ is a tautology}$$
$$\mathcal{B}((\mathcal{B}A \wedge \mathcal{B}(A \to B)) \to \mathcal{B}B)$$
$$\mathcal{B}(\mathcal{B}((\mathcal{B}A \wedge \mathcal{B}(A \to B)) \to \mathcal{B}B)$$
$$\mathcal{B}(\mathcal{B}A \to \mathcal{B}\mathcal{B}A)$$
$$\mathcal{B}\mathcal{B}(\mathcal{B}A \to \mathcal{B}\mathcal{B}A)$$

A reasoner *knows* that (s)he is of type X if (s)he is of type X and believes that (s)he is of that type.

Exercises/Theorems

1. Prove that if a reasoner of type 4 knows that (s)he is regular.
2. Prove that a reasoner of type 4 knows that (s)he is of type 4.
3. Prove that if a reasoner of type 4 ever believes that (s)he cannot be inconsistent, (s)he will become inconsistent.
4. Suppose a normal reasoner of type 1 believes a proposition of the form $p \leftrightarrow \neg\mathcal{B}p$. Then:
   (a) If (s)he ever believes p, then (s)he will become inconsistent.
   (b) If (s)he is of type 4, then (s)he knows that if (s)he should ever believe p then (s)he will become inconsistent–i.e., (s)he will believe the proposition $\mathcal{B}p \to \mathcal{B}\perp$.
   (c) If (s)he is of type 4 and believes that (s)he cannot be inconsistent, then (s)he will become inconsistent.

12.3 References

Smullyan, Raymond (1987) *Forever Undecided* Alfred A. Knopf Inc.

Chapter 13

The Logic of Knowledge

The logic of knowledge is a modal logic which is used as a tool for analyzing multi-agent systems - players in a poker game, processes in a computer network, or robots on an assembly line. *The following is based on a paper by Halpern.*

13.1 Syntax and Semantics

We introduce three new operators:

- $\mathcal{K}_i A$ - agent i knows A if A is true in all worlds agent i thinks possible
- $\mathcal{E}_G A$ - each agent in the group G knows A
- $\mathcal{C}_G A$ - A is common knowledge among the agents in the group G

The formulas of the logic of knowledge are

Let $\mathcal{P} = \{p_i | i = 0, 1, ...\}$ be a set of *atomic formulas* and $P \in \mathcal{P}$
Formulas

$F ::= P \mid \neg F \mid (F \vee F) \mid (F \wedge F) \mid (F \rightarrow F) \mid (F \leftrightarrow F)$ $F ::= P \mid \neg F \mid \wedge FF$ *The set of propositional formulas.*
$F ::= P \mid \mathbf{f} \mid \rightarrow FF$ *The set of propositional formulas.*
$F ::= P \mid \mathbf{f} \mid \rightarrow FF \mid \Box F$ *The set of propositional formulas.*
$F ::= P \mid \mathbf{f} \mid \rightarrow FF \mid \mathcal{B} F$ *The set of propositional formulas.*
$F ::= P \mid \mathbf{f} \mid \rightarrow FF \mid \mathcal{K}_i F \mid \mathcal{E}_G F \mid \mathcal{C}_G F$ *The set of propositional formulas.*

Common knowledge is defined as "everyone knows A, and everyone knows that everyone knows A, and ..."

The semantics for knowledge logic:

$\mathcal{M} = \langle \mathcal{W}, v \rangle$ where
$W = \{w_i \mid w_i \text{ is a relational structure}\}$
$K = \{K_i \mid K_i \subset w_i \times W \text{ is an accessiblity relation from } w_i\}$
$G = \{g_i \mid g_i \subset W\}$
$v_i : F \rightarrow \{0, 1\}$ – is a valuation function for w_i

not $M \models \mathbf{f}$	
$M \models P$	iff $v(p) = 1$
$M \models\rightarrow AB$	iff $M \models B$ or not $M \models A$
$M \models_u \mathcal{K}_i A$	iff $M \models_v A$ for all v such that $K_i uv$
$M \models \mathcal{E}_G A$	iff $M \models \mathcal{K}_i A$ for all $i \in G$
$M \models \mathcal{C}_G A$	iff $M \models \mathcal{K}_i A$ for all $i \in G$

The following axiom system is sound and complete.

Axioms

A1.	All propositional tautologies
A2.	$K_1 A \wedge K_i(A \rightarrow B) \rightarrow K_i B$
A3.	$K_i A \rightarrow A$
A4.	$K_i A \rightarrow K_i K_i A$
A4.	$\neg K_i A \rightarrow K_i \neg K_i A$
R1.	From A and $A \rightarrow B$ infer B
R2.	From A infer $K_i A$
C1.	$E_G A \rightarrow \forall(i \in G).K_i A$
C2.	$C_G A \rightarrow E_G(A \wedge CA)$
RC1.	From $A \rightarrow E_G(A \wedge B)$ infer $A \rightarrow C_G B$

A1 and R1 are a sound an complete axiom system of classical propositional logic. A2 says that an agent's knowledge is closed under implication. A3 says that an agent knows only things that are true. This axiom is usually taken to distinguish *knowledge* from *belief* i.e., false statements may be believed but not known. A4 and A5 are axioms of introspection; these are usually rejected by philosophers.

C2 (fixed point axiom) says that common knowledge of A holds exactly when everyone in the group knows A and that A is common knowledge.

13.2 Exercises

1. Rewrite the following in English: $K_1 K_2 A \wedge \neg K_2 K_1 K_2 A$.

2. Express symbolically, Dean doesn't know whether Nixon knows that Dean knows that Nixon knows that McCord burgled O'Brien's office at Watergate. Hint: let A be the statement "McCord burgled O'Brien's office at Watergate".

3. Express symbolically, Everyone in G knows p, but p is not common knowledge.

4. Construct a model for the situation where agent 1 does not know "it is sunny in San Francisco" but agent 2 does.

5. Interpret axiom A4: $K_i A \to K_i K_i A$

6. Interpret axiom A5: $\neg K_i A \to K_i \neg K_i A$.

7. Show that A3 - A5 are valid.

8. Show that C2 is necessary for agreement and coordination.

9. Show that the theory is decideable and decidability is of exponential complexity.

13.3 Multi-Agent Systems

13.4 Logical Omniscience

An agent is *logically omniscient* iff it knows all tautologies and its knowledge is closed under modus ponens.

What is logically knowable is not realizable in practice since real agents are *resource-bounded*. Attempts to define knowledge in the presence of bounds include

- restricting what an agent knows to a set of formulas which is not closed either under inference or all instances of the a given axiom.
- defining a set of possible worlds which in turn, defines a set of formulas.
- change the logic to a non-standard logic such as relevance logic
- a impossible worlds to the list of worlds
- restrict the depth of **K**s found in formulas
- add an operator for awareness so that the formulas that an agent is aware of is a subset of the formulas. An agent knows a formula iff it is true in each possible world of the agent.
- awareness can be defined to mean that an agent can use a local algorithm to compute an answer.

13.5 Future Directions

- Implement a logical agent.
- Reason about knowledge/belief change over time
- Knowledge based programming: The goal is to allow the programmer to write a program by saying what she wants rather than painfully describing how to compute what she wants.
- Analyze protocols and construct logical agents to implement the protocol
- More realistic models of knowledge that incorporate resource-bounded reasoning, probability, and the possibility of errors.
- A deeper understanding of the interplay between various modes of reasoning under uncertainty.

13.6 References

Halpern, Joseph Y *Reasoning about Knowledge: A Survey* (1995).

Chapter 14

Algorithmic Logic

Chapter 15

Lambda Calculus and Combinators

15.1 Functions

The classical idea of a function that of an optional name, 0 or more parameters, and a rule for computing a value from the parameters. For example,

$$f n = \text{if } (n == 0) \text{ then } 1 \text{ else } n * (f(n-1))$$

This is in contrast to the modern definition of a function as a set of ordered pairs. The classical idea is used as a foundation for computing. The lambda calculus is an elegant formulation of the classical idea.

Why should we be interested in the lambda calculus?

- The lambda calculus was introduced as a notation for representing and studying functions.
- As a model of computation, it is equivalent to the Turing machine.
- It provides the theoretical foundations for functional programming languages (Lisp is syntacticaly similar) and is used to provide the denotational semantics for programming languages.
- In it, functions are first class objects. They may be assigned to a variable, passed as an argument, or returned as a result.
- It is the source of inspiration for lexical scope rules and lazy evaluation.

What are the key ideas of the lambda calculus?

- Simple syntax
- Execution (expression evaluation) is by reduction to a normal form.

Syntax

The Syntax of the Lambda Calculus

formula	::=	term = term
term	::=	identifier | application | abstraction
identifier	::=	x_0, | x_1 | x_2 | ...
application	::=	(term term)
abstraction	::=	(λ identifier . term)

Parentheses may be omitted according to the following conventions:

1. Associate to the left: $ABC = ((AB)C)$
2. The scope of $\lambda x.$ extends as far a possible: $\lambda x.yx\lambda z.xz = (\lambda x.yx(\lambda z.xz))$
3. Consecutive λs may be collapsed to a single λ; $\lambda xy.M = \lambda x.\lambda y.M$

Substitution rules: $[e]^x_y$ – x replaces y in e.

$[x]^e_x$	$\rightarrow$	e	
$[y]^e_x$	$\rightarrow$	e	$x \neq y$
$[(M\ N)]^e_x$	$\rightarrow$	$([M]^e_x\ [N]^e_x)$	
$[\lambda x.M]^e_x$	$\rightarrow$	$\lambda x.M$	
$[\lambda y.M]^e_x$	$\rightarrow$	$\lambda z.[[M]^z_y]^e_x$	$x \neq y$ and z does not occur in either e or M

The Theory

Axioms and rules of inference

= is an equivalence relation

M=M
If M=N, then N=M
If M=N and N=L, then M=L

= is substitutive

If N=N', then M N = M N'
If M=M', then M N = M' N
If M=N, then λ x. M = λ x.N

Axioms - the conversion rules

alpha	$\lambda x.M$	=	$\lambda y.[M]^y_x$ (y does not occur in M
beta	$\lambda x.M$	=	$[M]^N_x$
eta	$\lambda x.M$	=	M

Reduction

Definition 15.1 *A* reduction *is the application of 0 or more alpha, beta or eta reductions.*
M is reduceable *to N (M* red *N) if there is a sequence of alpha, beta, and/or eta reductions from M to N.*
M is convertable *to N (M* cnv *N) if $M = N$ is a theorem of the lambda calculus.*
A term containing no free variables is called a combinator*.*

Definition 15.2 *A term M is said to be in* normal form *if it does not contain a (beta or eta) redux as a subterm. If M cnv N and N is a normal form, then N is said to be the* normal form of *M.*

The key questions include

- If two or more reductions are possible, does the order in which they are done matter? Some possibilities include left to right, right to left, parallel.
  - Left to right reductions correspond to lazy evaluation with significant overhead for copying but permit infinite data structures.
  - Right to left reductions correspond to passing by value.
- Do reduction sequences terminate?
- Do all functions reduce to a cononical form?

The following expressions do not have normal forms:

- $(\lambda x.xx)(\lambda x.xx)$
- $Y = \lambda f.(\lambda x.f(xx))(\lambda x.f(xx))$

The order of reduction is important:

$$(\lambda w.z)((\lambda x.xx)(\lambda x.xx))$$

Reducing the argument first is known as *applicative order* evaluation and corresponds to *call by value*. Performing the substitution first is known as *normal order* reduction and corresponds to *call by name*. Applicative order is desireable when a function is sure to use its arguments as the arguments are evaluated just once and may be used several times. Normal order reduction requires that the arguments be evaluated each time they are needed however, it is possible that the value of an argument may not be needed.

The following theorem assures us that any two reduction sequences that terminate will do so with the same result.

Theorem 15.1 *(Church-Rosser) If X ¡i¿cnv¡/i¿ Y then there is a Z such that X ¡i¿red¡/i¿ Z and Y ¡i¿red¡/i¿ Z.*

The Semantics

Combinators (variable-free expressions)

Haskell B. Curry

Combinatorial Logic

SKI & Y Combinator definitions

$$
\begin{array}{lcl}
\mathbf{S} & = & \lambda f.(\lambda g(\lambda x f x(gx))) \\
\mathbf{K} & = & \lambda x.\lambda y.x \\
\mathbf{I} & = & \lambda x.x \\
\mathbf{Y} & = & \lambda f \lambda x(f(xx))\lambda x(f(xx))
\end{array}
$$

Lambda expression to Combinator Translation

$$
\begin{array}{lcl}
C[s] & \rightarrow & s \\
C[(E_1\ E_2)] & \rightarrow & (C[E_1]\ C[E_2]) \\
C[\lambda x E] & \rightarrow & A[(x, C[E])] \\
A[(x, s)] & \rightarrow & \text{if } (s = x) \text{ then } \mathbf{I} \text{ else } (\mathbf{K}\ s) \\
A[(x, (E_1\ E_2))] & \rightarrow & ((\mathbf{S}A[(x, E_1)])A[(x, E_2)])
\end{array}
$$

Combinator Reduction Rules

$$
\begin{array}{lcl}
\mathbf{S}\ \mathrm{f\ g\ x} & \rightarrow & \mathrm{f\ x\ (g\ x)} \\
\mathbf{K}\ \mathrm{c\ x} & \rightarrow & \mathrm{c} \\
\mathbf{I}\ \mathrm{x} & \rightarrow & \mathrm{x} \\
\mathbf{Y}\ \mathrm{e} & \rightarrow & \mathrm{e}\ (\mathbf{Y}\ \mathrm{e}) \\
(\mathrm{A\ B}) & \rightarrow & \mathrm{A\ B} \\
(\mathrm{A\ B\ C}) & \rightarrow & \mathrm{A\ B\ C}
\end{array}
$$

Recursive Functions

15.1.1 References

Chapter 16

Computable

16.1 The Church-Turing Thesis

David Hilbert's tenth problem was to devise an algorithm that tests whether a polynomial has an integral root, i.e., find a process which determines the result in a finite number of operations. The search for a formal representation of "process", which we now call an algorithm, lead to a variety of formal models of computation which are all provably equivalent. Some of those models are:

- Turing machines (Alan Turing),
- Functional models
  - Partial Recursive Functions (Stephen Kleene),
  - Lambda Calculus (Alonzo Church),
- Grammar like rules
  - Post Formal Systems (Emile Post),
  - Markov Algorithms (Markov)
  - Unrestricted Grammars (N. Chomski),
- Recursively Enumerable Languages and intuitively,
- what is computable by a computer program written in any reasonable programming language.

Church's hypothesis, which is often called the

> **Church-Turing Thesis**, is a mathematically unprovable belief that any reasonable intuitive definition of "effectively computable" is equivalent to these formal models of computation.

This belief is based on the fact that these definitions were independently proposed as definitions of computable and later proved to be equivalent. Thus, the Church-Turing thesis suggests that these definitions are the best that we can do.

16.1.1 Turing Machines

The Turning machine was proposed to formalize the intuitive notion of an algorithm by Alan Turing in 1936. A Turing machine consists of an *infinite tape* of cells each of which can contain a symbol, a *tape head* that reads and writes symbols on the tape and moves left and right along the tape, and a *finite state control* (see Figure 16.1).

A basic Turing machine is a model for studying computation. Turing machines can solve decision problems and compute results based on inputs. When studying computation we usually restrict our attention to integers. Since a real number has infinitely many fraction digits we can not compute a real number in a finite time. Programming a Turing machine is tedious and thus much work at higher levels of abstraction make the reasonable assumption that any completely defined algorithm or computer program could be implemented by a Turing machine.

There are a lot of possible Turing machines and a useful technique is to code Turing machines as binary integers. A trivial coding is to use the 8 bit ASCII for each character in the written description of a Turing machine concatenated into one long bit stream. Having encoded a specific Turing machine as a binary integer, we can talk about TMi as the Turing machine encoded as the number "i". Since a description of a Turing machine is a finite description, the set of all Turing machines is countable and enumerable. We may enmerate (or list) all Turing machines by listing them in order of the number of states they have.

It is possible to construct a Universal Turing Machine, UTM, that takes an encoded Turing machine on its input tape followed by normal Turing machine input data on that same input tape. The Universal Turing Machine first reads the description of the Turing machine on the input tape and uses this description to simulate the actions that the encoded Turing machine would do when given the input data. Of course a UTM is a TM and can thus be encoded as a binary integer, so a UTM can read a UTM from the input tape, read a TM from the input tape, then read the input data from the input tape and proceed to simulate the UTM that is simulating the TM. Etc. Etc. Etc.

Since a UTM can be represented as an integer and can thus also be the input data on the input tape of itself or another Turing machine. This will be used below in the Halting Problem.

The Turing Machine Model is TM = $\langle Q, \Sigma, \Gamma, \delta, q_0, B, F \rangle$

Q	=	finite set of states $\{q_0, q_1, ..., q_n\}$ for some n in $\mathbb{N}$
Σ	=	finite set of input symbols not including B
Γ	=	finite set of tape symbols including those in Σ and the symbol B
δ	=	a finite set of transitions mapping $Q \times \Gamma$ to $Q \times \Gamma \times \{L, R\}$
q_0	=	initial state
B	=	blank tape symbol, initially on all tape not used for input
F	=	set of final states

```
+-------------------------+-----------------
| input string            |BBBBB ...  accepts Recursively Enumerable Languages
+-------------------------+-----------------
   ^ read and write, move left and right
   |
   |  +-----+
   |  |     |--> accept
   +--+ FSM |
      |     |--> reject
      +-----+

+-------------------------+-----------------
| input and output string |BBBBB ...  computes partial recursive functions
+-------------------------+-----------------
   ^ read and write, move left and right
   |
   |  +-----+
   |  |     |
   +--+ FSM |--> done  (a delta [q,a]->[empty], but may never happen )
      |     |
      +-----+
```

δ may be represented as a table or a list of the form:
$[q_i, a_i] \rightarrow [q_j, a_j, L]$ or $[q_i, a_i] \rightarrow [q_j, a_j, R]$
where
q_i is the present state
a_i is the symbol under the read/write head
q_j is the next state
a_j is written to the tape at the present position
L is move the read/write head left one position after the write
R is move the read/write head right one position after the write

Figure 16.1: Defintion of a Turing Machine

Intuitive notion of algorithms	equals	Both Turing machine and λ-calculus algorithms

Figure 16.2: The Church-Turing Thesis

The Turing Machine Hierarchy

$TM = \langle Q, \Sigma, T, q_0, q_{\text{accept}}, q_{\text{reject}} \rangle$

1. Q is a finite set of states.
2. Σ is a finite input alphabet not containing the special blank symbol $\llcorner\!\lrcorner$.
3. T is the finite tape alphabet, where $\{\llcorner\!\lrcorner\} \cup \Sigma \subset T$.
4. $\delta : Q \times T \to Q \times T \times \{\text{L, R}\}$ is the transition function of the finite control.
5. $q_0 \in Q$ is the start state.
6. $q_{\text{accept}} \in Q$ is the accept state.
7. $q_{\text{reject}} \in Q$ is the reject state where $q_{\text{reject}} \neq q_{\text{accept}}$.

$\delta(q, a) = (r, b, D)$ – when the machine is in the state q and the head over tape symbol a, the machine writes the symbol b replacing a, goes to state r, and moves the head one cell in the direction D.

Church-Turing thesis: David Hilbert's tenth problem was to devise an algorithm that tests whether a polynomial has an integral root i. e., find a process which determines the result in a finite number of operations. Both the λ-calculus (devised by Alonzo Church) and the Turing machine capture this intuitive notion of an algorithm.

Definition 16.1 *A language is* Turing-recognizable *(recursively enumerable) if some Turing machine recognizes it.*

Recognition means that the TM halts in an accepting state for elements of the language and either halts in a rejecting state or fails to halt for elements not in the language.

Definition 16.2 *A language is* Turing-decidable *(recursive) or simply* decidable *if some Turing machine decides it. Decidable means that the TM always halts.*

Theorem 16.1 *A language is Turing-recognizable if and only if some enumerator enumerates it.*

Theorem 16.2 *Some languages are not Turing-recognizable (non-computable function).*

> *Proof Outline:*
> Show that Σ^* is countable.
> Show that the set of TMs is countable.
> Show that B, the set of infinite strings over 0, 1, is uncountable by diagonalization.
> Show that L, the set of all languages over $\Sigma*$ (i.e. $L = 2^{\Sigma*}$), is uncountable by a correspondence with B.
> Therefore, there are languages which are not Turing-recognizable.

Enumerator

An enumerator is a TM (with an attached printer) that outputs the elements of a set.

Theorem 16.3 *A language is Turing-recognizable iff some enumerator enumerates it.*

Gödel Numbering

The Halting Problem

Related to Gödel's incompleteness problem is the Halting Problem for Turing machines. The halting problem is to determine for any given Turing machine and any input, whether the Turing machine halts in either an accepting or rejecting state.

Theorem 16.4 *The halting problem is undecidable.*

Definition 16.3 Decidable: *A set S is said to be decidable if there is a process P such that for $\forall x.P(x) = x \in S$.*

Theorem 16.5 *The language*

$$Accept = \{\langle m, w\rangle \mid m \textit{ is a } TM, \ w \textit{ is a string, and } m(w) = accept\}$$

is not decidable.

> *Proof:* By contradiction. Assume that *Accept* is decidable, that is, let H be a TM such that $H(\langle m, w\rangle) =$ accept if $m(w) =$ accept and otherwise, $H(\langle m, w\rangle) =$ reject.
>
> Let D be a TM such that for every TM, M, $D(M) =$ accept if $H(\langle M, M\rangle) =$ reject and $D(M) =$ reject if $H(\langle M, M\rangle) =$ accept.

Now we have a contradiction since D(D) = accept if D(D) = reject and D(D) = reject if D(D) = accept. Thus the language *Accept* must not be decidable. QED.

i.e. D(D) = accept if $H(\langle D, D\rangle)$ = reject (i.e. D(D) = reject) and D(D) = reject if $H(\langle D, D\rangle)$ = accept (i.e., D(D) = accept).

Universal TM (UTM)

It is possible to code the definition of every Turing machine as a string of characters. A Universal Turing Machine is then a Turing machine that has on its input tape a definition of some Turing machine and a string. The Universal Turing Machine simulates the Turing machine defined on its input tape and accepts the string on its input tape if and only if the simulated Turing machine would accept the string.

One UTM can simulate another UTM that is simulating a TM with an input string. Etc. Etc. Etc.

Self reproducing TMs

Nondeterministic TMs

A nondeterministic Turing machine has a transition relation rather than a transition function. For example:

$$\delta(q, a) = \{(r_i, b_i, D_i), (r_j, b_j, D_j), ...\}$$

Theorem 16.6 *Every nondeterministic TM has an equivalent deterministic TM.*

Theorem 16.7 *A language is decidable iff some nondeterministic Turing machine decides it.*

Probabilistic TMs

A probabilistic TM is a nondeterministic TM where each nondeterministic step has two legal next moves.

Alternating TMs

Oracle TMs

Quantum TMs

Input tape cells are quantum states. Internal states are quantum states, transitions are quantum transitions.

A Quantum Turing Machine, as defined by Bernstein and Vazirani, is a triplet $\langle \Sigma, Q, \delta \rangle$, where Σ is a finite alphabet with an identified blank symbol #, Q is a finite set of states with an identified initial state q_0 and final state q_f not the same as q_0. δ is the transition function

$$\delta : Q \times \Sigma \to \mathbb{C}^{\Sigma \times Q \times \{L,R\}}$$

where $\mathbb{C}$ is the field of complex numbers. The QTM has an infinite tape of cells indexed by $\mathbb{Z}$ and a single read/write tape head. This basically means that δ yields a set of elements of $\Sigma \times Q \times \{L, R\}$ with a complex amplitude for each element.

A *configuration* is defined identically to the familiar definition for a classical TM: a description of the contents of the tape, the location of the tape head, and the state q. Thus, a QTM is in a superposition of configurations, not just a single configuration. A *final configuration* is a configuration in state q_f. We say that QTM M halts with running time T on input x if, at time T the superposition contains only final configurations.

We immediately see how the QTM is founded on quantum mechanics: the machine is allowed to be in a superposition of "states", which are, in this case, configurations. This seems to be a form of computational parallelism.

`../CII/wicked.html` Computational complexity and problem hierarchy

16.1.2 Lambda Calculus

Also known as Church's Lambda Calculus, we describe here the non pure form which has constants.

An expression in the Lambda Calculus, E, is defined as

E ::= C | V | (E E) | (λ v E) | (E) where

C is a set of constants including untyped values and function names
V is a finite set of variables
E defines a set of expressions called lambda expressions.

A Lambda Expression is one of the following:

1. a constant from C
2. a variable from V
3. a combination (E_i E_j) involving the application of expression E_i to another expression E_j. E_i is the operator and E_j is the operand.
4. an abstraction involving a variable v from V and an expression E_i. λ v E_i where v is called the bound variable and E_i is the body.

Parentheses, (E_i), are not really needed but are allowed to make it easier to read Lambda Expressions; A B C ... K means ((...((A B) C) ...) K). A period may also be used to separate the bound variable from the body as λv.E.

A *beta* reduction is (λ v.A c $\rightarrow$ A[c/v] which says c is substituted for all free occurrences of v in A.

An *eta* reduction is λ v.E v $\rightarrow$ E provided v is not free in E.

Example: (λ x.plus x 1)((λ y.times y y)3) $\Rightarrow$ plus(times 3 3)1 which equals 10.

Various orders of evaluation are possible, one of the most used is leftmost-outermost (normal form order)(safe)

For more information see ”¡a href=”../Logic/lambdaCalculus.html”¿Lambda Calculus and Combinators¡/a¿”

Post Formal Systems

This is somewhat similar to formal grammars yet is has an easier conversion to Turing machines and uses the concept of axioms (see Figure 16.3).

Partial recursive functions

A Partial Recursive Function is allowed to have an infinite loop for some input values. A Recursive Function also called a Total Recursive Function always returns a value for all possible input values.

Partial Recursive Functions correspond to Turing machines that may not halt. (Total) Recursive Function correspond to Turing machines that always halt.

Primitive Recursive Functions are a subset of Total Recursive Functions with the restriction that only primitive recursion is used a finite number of times and recursion uses zero and the successor function.

Primitive recursion is defined for $f(x_1, ..., x_n)$ as $f(x_1, ..., x_n) = g(x_1, ..., x_{n-1})$ if $x_n = 0 = h(x_1, ..., x_n, f(x_1, ..., x_{n-1}, x_{n-1}))$ if $x_n > 0$ where g and h are primitive recursive functions.

Ackermann's function is not primitive recursive.

For technical reasons a projection function, a selector, is often used. $\mathrm{P}_i(\mathrm{x}_1,...,\mathrm{x}_n)$ returns x_i where $1 \leq i \leq n$.

y = $\mathrm{f}(\mathrm{x}_1,...,\mathrm{x}_n)$ is a partial recursive function over the natural numbers when f is defined by a finite set of rules using a finite set of variables and a finite set of constants from the set of natural numbers. The function f can make use of itself and other partial recursive functions. A typical base function is the successor function, add one, and the constants typically include zero. The arguments $\mathrm{x}_1,...,\mathrm{x}_n$ and the result value y are from the set of natural numbers.

This can be extended to partial recursive functions over the integers and over the rational numbers, ratio of two integers, but can not be extended to the set

A Post System Pi = (C, V, A, P) where
C = {non terminal constants} union {terminal constants}
V = {finite set of variables}
A = {a finite set from C*, called the axioms} P = finite list of productions of the form: x1 v1 x2 ... xn vn xn+1 → y1 w1 y2 ... yn wn yn+1 where
xi and yi are in C*
vi and wi are in V with restriction wi ≠ wj saying that a variable may appear at most once on the left
union wi is a subset of union vi saying that each variable wi on the right must appear on the left

Post Systems can express arithmetic, as in the example:

```
Ct = { 1, +, = }
Cn = Phi
V  = {v1, v2, v3}
A  = { 1 + 1 = 11 }    tally notation for one plus one equals two

P1   v1 + v2 = v3  ->  v1 1 + v2 = v3 1
P2   v1 + v2 = v3  ->  v1 + v2 1 = v3 1

The example system allows the derivations

 1 + 1 = 11  =>  11 + 1 = 111         by P1  1+1=2 => 2+1=3
             =>  11 + 11 = 1111       by P2  2+2=4
             =>  111 + 11 = 11111     by P1  3+2=5
             =>  111 + 111 = 111111   by P2  3+3=6     etc.
```

Figure 16.3: Post System

of real numbers. y=f(x) is not a partial recursive function when x and y are from the set of real numbers and f(x) is defined as the square root of x, also written as the value of y that satisfies y**2 = x or y**2 - x = 0. For example, when x = 2, y is the square root of 2 which can not be computed in a finite time and yet is a well defined value for a real number.

Unrestricted grammars

A grammar is another way to pose a ¡a href=”#decision”¿decision problem.¡/a¿ A grammar takes a string as input and accepts or rejects the string. Thus a grammar characterizes a language, usually written L(G). ¡a href=”lang_def.html#Grammar”¿More details about grammars.¡/a¿

16.2 Formal Languages

A formal language is defined as set of finite strings over an alphabet of finite symbols. A decision problem can be posed as given a language is a given string in the language. Basically this is the mathematical problem of given a set is a particular element in the set.

¡a href=”lang_def.html#Language”¿More details about formal languages.¡/a¿

16.3 Functions

A function is a mapping from a domain to a range. A *total function* outputs something for every input. A *partial function* may to produce an output for only some inputs. By grouping the inputs, any function with more than one input can be represented by a function with only one input. Ditto for output.

A computable function may be expressed in many forms, yet to be reasonable, a description of a function as a rule form computation must be finite. Thus all functions can be expressed as a subset of Σ^* for some finite alphabet Σ. As the size of Σ^* is countable, the set of all computable functions is countable.

The range of a function can be a subset of real numbers, but the real numbers are uncountable, thus there are real numbers not computable by any function. A mapping can be defined for which there is no computable function.

Functions are some times categorized by the relations of the domain and range. The terms injection, surjection and bijection are all total functions defined as:

Injection: one-to-one, into: for every member of the domain, the function returns some member of the range, but not necessarily all members of the range will be returned.

Surjection: many-to-one, onto: for every member of the domain, the function returns some member of the range. Every member of the range is returned for some member of the domain, but unique members of the domain may return the same member of the range.

Bijection: one-to-one, onto: for every member of the domain, the function returns a unique member of the range.

Inverse: The inverse function of some function has the domain and range interchanged. The inverse of a Bijection is a Bijection. Inverse functions do not exists for general Injection or Surjection functions.

Example: y = sin(x). The computer approximation of the trigonometric sine function, y is a member of the Range -1.0 .. +1.0, the floating point values x is a member of the Domain of all floating point values. sin is a Surjection function, many-to-one onto.

16.4 Computational Complexity and Problem Hierarchy

Space, Time, and Problem Definition. There are three dimensions of problem complexity, time, space, and politics. In computer science, computational complexity refers to the first two dimensions, time and space. Problems having a social aspect often have a political dimension that requires compromize for solution rather than appeal to physical properties.

The running time and the space requirements of a program are defined as a function of the size of the input to the program. So, for example, an algorithm for sorting a list of n items might be described as requiring a running time of order n^2 and space of order n. Programs of order n^k (k is a constant) or less are said to be of polynomial complexity and are typical of the programs run on sequential computers. Programs of order k^n (k is a constant) are of exponential complexity and require more resources (except for small n) than are available. Problems of exponential complexity are "solved" using approximate methods that are of polynomial complexity. Theorem proving is an example of a problem of exponential complexity while proof checking is of linear (polynomial) complexity. If there are an unlimited number of computers, then some exponential complexity problems may be solved in polynomial time. Problems which require exponential resources are said to be *intractable*. The problems discussed so far are all computable functions. The focus of the remainder of this subsection is on time complexity.

¡a href="complexity.html"¿For more details about computational complexity.¡/a¿

16.5 Decision Problems

Given a sentence S of logic, we would like to know whether or not it is true, i.e., $\models S$ and whether or not it is a theorem, i.e., $\vdash S$. These questions are examples of decision problems.

Decision problems are stated as questions where the answer is binary, 0 or 1, False or True, No or Yes, Reject or Accept and so forth. Generally a decision problem states a problem and gives a candidate solution, asking if the solution solves the problem.

Examples:

- Given the math expression 2+2 is the answer 4?
- Given a formal language and a string, is the string in the language?
- Given a grammar and a string, is the string accepted by the grammar?
- Given a sentence S, if S is true, $\models S$, is it decidable whether or not it is provable, $\vdash S$?
- Given a sentence S, if S is a theorem, $\vdash S$, is it decidable whether or not it is true, $\models S$?
- Given a sentence S, is it decidable whether or not it is true, $\models S$ or $\not\models S$?
- Given a sentence S is it decidable whether or not it is a theorem, $\vdash S$ or $\nvdash S$?
- Given a Turing machine, TM, and a number, n, is it decidable whether or not TM halts for n?

A *positive solution* to a decision problem requires the construction of an algorithm which solves the problem. A *negative solution* to a decision problem requires that it be shown that there is no algorithm to solve the problem. Such problems are called *unsolvable*. This requires a satisfactory definition of an algorithm.

In the case of logic, the soundness theorem tells us that for all sentences S, if S is a theorem, $\vdash S$, then S is true, $\models S$. Furthermore, the theorems can be enumerated by systematically constructing proofs of increasing length. However, there is no algorithm to show that a sentence S is not a theorem, i.e., $\nvdash S$. But soundness alone does not provide a decision method for truth. Gödel's completeness theorem for inference rules tells us that for all sentences S, if S is a theorem, then there is an algorithm to construct its proof. Again, completeness alone does not provide a decision method for theoremhood. Now Gödel's incompleteness theorem tells us that there is a true sentence S, $\models S$, but it not a theorem, $\nvdash S$. Thus logic does not provide a decision method for either truth or theoremhood.

The decision problem in mathematics is to determine whether various propositions concerning mathematical objects are true or false. For this task, mathematicians employ various methods of proof. Gödel's incompleteness theorem describes limits what mathematicians can do. Gödel showed that there are propositions which are true but which cannot be proved. The finite methods of proof are unable to reach all true statements.

The decision problem in computer science is to determine whether there is an algorithm or effective computational procedure for solving various problems. For this task, computer scientists employ various definitions of algorithms. The negative solution to the Halting Problem demonstrates that there are limitations on what computer scientists can do. The Halting Problem shows that there are problems which cannot have an algorithmic solution.

Definable

Definition: x is definable iff there is a predicate P(t) such that t is the only t such that P(t). Implicit is the notion that the definition is finite.

Definable is not the same as computable:

> Let x be the real number whose n-th digit is 0 iff the n-th Turing machine stops and 1 otherwise. x is defined but not computable.

Unsolvable

A formally stated problem is solvable if a Turing machine exists to compute the solution. A formally stated problem is provably solvable if it can be proved that Turing machine exists to compute the solution.

A formally stated problem is *unsolvable* if no Turing machine exists to compute the solution. A formally stated problem is provably unsolvable if it can be proved no Turing machine exists to compute the solution.

Undecidable

Solvability is one half of the decision problem. Simply put, the decision problem is to determine whether or not a problem is solvable.

A formally stated problem is decidable if there is total recursive function and thus, a Turing machine that always halts, that can be constructed to decide whether the problem is true or false. In terms of solvability, a decidable problem is one for which it can be determined whether or not it is solvable.

A formally stated problem is *undecidable* if no total recursive function and thus, no Turing machine that always halts, can be constructed to decide the problem.

The halting problem is to produce an algorithm which will determine for any algorithm whether or not the algorithm will halt on any given input.

Figure 16.4: The Halting Problem

Gödel’s Incompleteness Theorem

The decision problem for any logical system is whether or not the system is able to determine for an arbitrary proposition whether or not it is provable. Gödel’s completeness theorem for first-order logic states that there is a method to produce proofs of those statement that are provable. This is the first half the decision problem for logic. The Halting problem shows that only the first half of the decision problem for logic exists.

Gödel’s Incompleteness Theorem states that any formal system powerful enough to express arithmetic must have true theorems that can not be proven within the formal system.

Basically Gödel proved that when trying to add axioms to a formal system in order to prove all true theorems within the formal system, eventually the system will become inconsistent before is becomes complete.

A complete formal system is a formal system where all true theorems can be proved.

An inconsistent formal system is a formal system where at least one false statement can be proved within the formal system.

Due to the computational equivalence of formal systems to other computational capability, we get the Halting problem, the uncomputable numbers and other unsolvable problems.

¡a href=”../Logic/General.html”¿More details about Gödel’s incompleteness theorem.¡/a¿

The Halting Problem

The “Halting Problem” is a very strong, provably correct, statement that no one will ever be able to write a computer program or design a Turing machine that can determine if a arbitrary program will halt (stop, exit) for a given input.

This is NOT saying that some programs or some Turing machines can not be analyzed to determine that they, for example, always halt.

The Halting Problem says that no computer program or Turing machine can determine if ALL computer programs or Turing machines will halt or not halt on ALL inputs. To prove the Halting Problem is unsolvable we will construct one program and one input for which there is no computer program or Turing machine.

We will use very powerful mathematical concepts and do the proofs for both a computer program and a Turing machine. The mathematical concepts we need are:

Proof by contradiction. Assume a statement is true, show that the assumption leads to a contradiction. Thus the statement is proven false.

Self referral. Have a computer program or a Turing machine operate on itself, well, a copy of itself, as input data. Specifically we will use diagonalization, taking the enumeration of Turing machines and using TMi as input to TMi.

Logical negation. Take a black box that accepts input and outputs true or false, put that black box in a bigger black box that switches the output so it is false or true respectively.

The simplest demonstration of how to use these mathematical concepts to get an unsolvable problem is to write on the front and back of a piece of paper "The statement on the back of this paper is false." Starting on side 1, you could choose "True" and thus deduce side 2 is "False". But staring on side 2, which is exactly the same as side 1, you get that side 2 is "True" and side 1 is "False." Since side 1, and side 2, can be both "True" and "False" there is a contradiction. The problem of determining if sides 1 and 2 are "True" or "False" is unsolvable.

The Halting Problem for a programming language

We will use the "C" programming language, yet any language will work.

Assumption: There exists a way to write a function named `Halts` such that:

```
int Halts(char * P, char * I)
{
  /* code that reads the source code for a "C" program, P,
     determines that P is a legal program, then determines if P
     eventually halts (or exits) when P reads the input string I,
     and finally sets a variable "halt" to 1 if P halts on input I,
     else sets "halt" to 0 */
  return halt;
}
```

Construct a program called `Diagonal.c` as follows:

```
int main()
{
  char I[100000000];  /* make as big as you want or use malloc */
  read_a_C_program_into( I );
  if ( Halts(I,I) )  { while(1){} }   /* loop forever, means does not halt */
  else return 1;
}
```

Compile and link `Diagonal.c` into the executable program `Diagonal`. Now execute `Diagonal < Diagonal.c`.

Consider two mutually exclusive cases:

Case 1: `Halts(I,I)` returns a value 1. This means, by the definition of `Halts`, that `Diagonal.c` halts when given the input `Diagonal.c`. BUT! we are running `Diagonal.c` (having been compiled and linked) and so we see that `Halts(I,I)` returns a value 1 causes the "if" statement to be true and the "`while(1)`" statement to be executed, which never halts, thus our executing `Diagonal.c` does NOT halt. This is a contradiction because this case says that `Diagonal.c` does halt when given input `Diagonal.c`. Well, try the other case.

Case 2: `Halts(I,I)` returns a value 0. This means, by the definition of halts, that Diagonal.c does NOT halt when given the input Diagonal.c. BUT! we are running Diagonal.c (having been compiled and linked) and so we see that Halts(I,I) returns a value 0 causes the "else" to be executed and the main function halts (stops, exits). This is a contradiction because this case says that Diagonal.c does NOT halt when given input Diagonal.c. There are no other cases, Halts can only return 1 or 0. Thus what must be wrong is our assumption "there exists a way to write a function named Halts..." .

The Halting Problem for Turing machines

Assumption: There exists a Turing machine, TMh, such that: When the input tape contains the encoding of a Turing machine, TMj followed by input data k, TMh accepts if TMj halts with input k and TMh rejects if TMj is not a Turing machine or TMj does not halt with input k.

Note that TMh always halts and either accepts or rejects. Pictorially TMh is:

```
+----------------------------
| encoded TMj B k BBBBB ...
+----------------------------
    ^ read and write, move left and right
    |
    |  +-----+
    |  |     |--> accept
    +--+ FSM |                      always halts
       |     |--> reject
       +-----+
```

We now use the machine TMh to construct another Turing machine TMi. We take the Finite State Machine, FSM, from TMh and

1. make none of its states be final states

2. add a non final state ql that on all inputs goes to ql

3. add a final state qf that is the accepting state

Pictorially TMi is:

```
+-------------------------------------------
| encoded TMj B k BBBBB ...
+-------------------------------------------
   ^ read and write, move left and right
   |
   |  +----------------------------------+
   |  |                     __           |
   |  |                   /   \ 0,1      |
   |  |                 +-| ql |--+      |
   |  | +------+        | \___/   |      |
   |  | |      |--> accept-+   ^     |      |
   +--+-+ FSM |              |_____|      |            may not halt
      | |      |--> reject-+    _         |
      | +------+          |  // \\       |
      |                   +-||qf ||------|--> accept
      |                      \\_//       |
      +----------------------------------+
```

We now have Turing machine TMi operate on a tape that has TMi as the input machine and TMi as the input data.

```
+-------------------------------------------
| encoded TMi B encoded TMi BBBBB ...
+-------------------------------------------
   ^ read and write, move left and right
   |
   |  +----------------------------------+
   |  |                     __           |
   |  |                   /   \ 0,1      |
   |  |                 +-| ql |--+      |
   |  | +------+        | \___/   |      |
   |  | |      |--> accept-+   ^     |      |
   +--+-+ FSM |              |_____|      |            may not halt
      | |      |--> reject-+    _         |
      | +------+          |  // \\       |
      |                   +-||qf ||------|--> accept
      |                      \\_//       |
      +----------------------------------+
```

Consider two mutually exclusive cases:

Case 1: The FSM accepts thus TMi enters the state ql. This means, by the definition of TMh that TMi halts with input TMi. BUT! we are running TMi on input TMi with input TMi and so we see that the FSM accepting causes TMi to loop forever thus NOT halting. This is a contradiction because this case says that TMi does halt when given input TMi with input TMi. Well, try the other case.

Case 2: The FSM rejects thus TMi enters the state qf. This means, by the definition of TMh that TMi does NOT halt with input TMi. BUT! we are running TMi on input TMi with input TMi and so we see that the FSM rejecting cause TMi to accept and halt. This is a contradiction because this case says that TMi does NOT halt when given input TMi with input TMi. There are no other cases, FSM either accepts or rejects. Thus what must be wrong is our assumption “there exists a Turing machine, TMh, such that...” QED.

Thus we have proved that no Turing machine TMh can ever be created that can be given the encoding of any Turing machine, TMj, and any input, k, and always determine if TMj halts on input k.

Other Links

- ¡a href=”lang_def.html”¿Formal Language related definitions¡/a¿
- ¡a href=”automata_def.html”¿Automata related definitions¡/a¿
- ¡a href=”classes.html”¿Language class definitions¡/a¿

¡a href=”#Top”¿Go to top¡/a¿

Part III

Limits of Logic

Chapter 17

The Impossible

17.1 The Counter Intuitive

17.2 Skolem's Paradox

17.3 Impossible Constructions

17.4 Independence Results

¡ul¿ ¡li¿1+1=2 and not even God can change it.¡/li¿ ¡li¿Space, the final frontier ¡ul¿ ¡li¿Illusion¡/li¿ ¡li¿n-Dimensional Euclidian space¡/li¿ ¡li¿Counter Intuition ¡ul¿ ¡li¿Mbius' band and Klein's bottle¡/li¿ ¡li¿Space filling curves - fractals in general¡/li¿ ¡/ul¿ ¡/li¿ ¡/ul¿ ¡/li¿ ¡li¿Why can't we figure it out? ¡ul¿ ¡li¿Why Adam couldn't finish - Skolem's paradox.¡/li¿ ¡li¿Not with these tools. ¡ul¿ ¡li¿The trisection problem¡/li¿ ¡li¿Squaring the circle¡/li¿ ¡li¿Doubling the cube¡/li¿ ¡/ul¿ ¡/li¿ ¡li¿A Declaration of Independence ¡ul¿ ¡li¿Just which world did you say we live in? - The Parallel Postulate.¡/li¿ ¡li¿Which infinity is next? - The generalized continuum problem.¡/li¿ ¡/ul¿ ¡/li¿ ¡/ul¿ ¡/li¿ ¡/ul¿

Chapter 18

Gödel Numbers and Gödel Numbering

Numbers correlated to linguistic objects are called *Gödel numbers* and the correlation is called a *Gödel numbering*, after Kurt Gödel, who introduced it in 1931. The correlation must be such that both it and its inverse are effectively computable. Gödel numbering is often used to show that the elements of some set are countable, effectively constructable (recursively enumerable), and to implement self-reference for the purpose of demonstrating that something is not constructable (incompleteness or non-computability).

Let g be a 1-1 function, the Gödel numbering, which assigns to each expression E a natural number $g(E)$ called the *Gödel number* of E. We let E_n be that expression whose Gödel number is n. Thus, $g(E) = E_n$.

Not every number may be the Gödel number of an expression. However, this property may be ...

Let $\mathcal{E}$ be a denumerable set of expressions ...

A function $\mathcal{G}$ that assigns to every expression E and every natural number n an expression $E(n)$. The function obeys the condition that for every predicate H and every natural number n, the expression $H(n)$ is a sentence.

A function g is a Gödel numbering function if it satisfies the following:

1. g is 1-to-1

2. g is effectively computable

3. g^{-1} is effectively computable

18.1 Gödel Numbering for polynomial equations in one variable with integral coefficients

The set of polynomial equations in one variable with integral coefficients consist of combinations of the following fourteen symbols.

$$0\,1\,2\,3\,4\,5\,6\,7\,8\,9\,x\;+\;-\;=$$

The fourteen symbols may be regarded as the digits in a base fourteen number system. In this way, each equation is a number in that system. Using this scheme, the set of polynomial equations may be enumerated by systematically counting in base 14 and for each number, determining whether or not it is the number of an equation. Kleene refers to this type of Gödel numbering as the "method of digits" in his book, *Mathematical Logic*.

18.2 Gödel Numbering for a First-order Theory

The first-order theory K is constructed using the symbols.

$$(\;)\;,\;\neg\;\rightarrow\;\forall\;x_k\;a_k\;f_k^n\;A_k^n$$

where $k, n > 0$.

The formation rules for the theory K are:

Definition: An arithmetization of a first-order theory K is a one-to-one function G from the set of symbols of K, expressions of K, finite sequences of expressions of K, into the set of Natural numbers such that the function G and its inverse function are effectively computable, i.e., the following conditions are satisfied:

- G is effectively computable.
- There is an effective procedure that determines whether any give positive integer m is in the range of G, and if m is in the range of G, the procedure finds the object x such that G(x)=m.

Definition: The Theory K has odd Gödel numbers assigned in the following way

$G(() = 3; G()) = 5; G(,) = 7; G(\neg) = 9; G(\rightarrow) = 11; G(\forall) = 13;$

$G(x_k) = 7 + 8k$ for $k > 0$

$G(a_k) = 9 + 8k$ for $k > 0$

$G(f_k^n) = 11 + 8k$ for $k, n > 0$

$G(A_k^n) = 13 + 8k$ for $k, n > 0$

Each symbol as ...

Definition: Given an expression (i.e., a sequence of symbols) $u_0 u_1 ... u_r$, its Gödel number is even and prime factorization has odd exponents and is defined to be:

$$G(u_0 u_1 ... u_r) = p_1^{G(u_0)} p_2^{G(u_1)} ... p_r^{G(u_r)}$$

where the p_i is the ith prime. By the unique factorization of natural numbers into primes we know that different expressions have different Gödel numbers.

Definition: If we have an arbitrary finite sequence of expressions $e_0, ..., e_r$ its Gödel number is even and its prime factorization has even exponents and is defined to be:

$$G(e_0, e_1, ..., e_r) = p_1^{G(e_0)} p_2^{G(e_1)} ... p_r^{G(e_r)}$$

where p_i is the ith prime. By the unique factorization of natural numbers into primes we know that different expressions have different Gödel numbers.

18.3 Gödel Numbering for Turing Machines

A Turing Machine is defined by a set of transition functions of the following form:

$$[s_i, a_i][s_j, a_j, D]$$

The method of digits may be used to produce an enumeration of Turing machines.

18.4 Exercises

1. Construct a Gödel numbering for the language of
   (a) group theory.
   (b) finite automata.
   (c) pushdown automata.
   (d) Lambda Calculus.

Chapter 19

General Setting for Incompleteness

19.1 The General Idea Behind Gödel's Proof

Key Question. What are the sufficient conditions for a system in which each provable statement is true and every refutable statement is false, to contain true but unprovable sentences?

Abstract forms of Gödel's and Tarski's Theorems

Note: A predicate is a statement which defines the elements of a set. In this expression, $\{x|P(x)\}$, P is a predicate and the constants which satisfy it (make it true) are the elements of the set. Thus we can say the predicates define or name sets.

The languages to which Gödel's argument is applicable contain at least the following.

1. $\mathcal{L}$ is a subset of Σ^* called a *language*. Where,
   - Σ is a countable set of *symbols* and
   - Σ^* is the set of all *strings* of symbols of Σ.
2. $\mathbf{E}$ is a subset of $\mathcal{L}$ whose elements are called the *expressions* of $\mathcal{L}$.
3. $\mathbf{S}$ is a subset of $\mathbf{E}$ whose elements are called the *sentences* of $\mathcal{L}$.
4. $\mathbf{P}$ is a subset of $\mathbf{S}$ whose elements are called the *provable* sentences of $\mathcal{L}$.
5. $\mathbf{R}$ is a subset of $\mathbf{S}$ whose elements are called the *refutable* sentences of $\mathcal{L}$.

6. Φ is a function that assigns to every expression e of **E** and every natural number n of $\mathbb{N}$ an expression $e(n)$ i.e., $\Phi(e, n) = e(n)$ and $e(n) \in \mathbf{E}$. The function Φ satisfies the condition that for every *predicate*, H, and every natural number, n, the expression $H(n)$ is a sentence i.e, $\Phi(H(n)) \in \mathbf{S}$; $H(n)$ expresses the proposition that n belongs to the set named by H.

7. **T** is a subset of **S** whose elements are called the *true* sentences of $\mathcal{L}$.

The following relationships hold among the sets: **T**, **R**, and $\mathbf{P} \subset \mathbf{S} \subset \mathbf{E} \subset \mathcal{L} \subset \Sigma^*$.

Definition 19.1 Expressibility *in* $\mathcal{L}$. *By the set expressed by* H*, we mean the set of all* n *that satisfy* H*. Thus for any set* A *of numbers,* H *expresses* A *if and only if for every number* $n : H(n) \in T \leftrightarrow n \in A$

Definition 19.2 *A set* A *is called* expressible *or* nameable *in* $\mathcal{L}$ *if* A *is expressed by some predicate of* $\mathcal{L}$*.*

Theorem 19.1 *Some sets are not expressible (nameable).*

Proof. *Since there are only countably many expressions in* $\mathcal{L}$*, there are only finitely or countably many predicates of* $\mathcal{L}$*. But there are uncountably many sets of natural numbers (Cantor). Therefore, not every set of numbers is expressible in* $\mathcal{L}$*.*

Definition 19.3 *The system* $\mathcal{L}$ *is called correct if every provable sentence is true and every refutable sentence is false (not true). That is,* **P** *is a subset of* **T** *and* **R** *is disjoint from* **T** *(i. e.* $\mathbf{P} \subset \mathbf{T}$ *and* $\mathbf{R} \cap \mathbf{T} = \emptyset$*).*

Gödel numbering and diagonalization.

Let g be a 1-1 function which assigns to every expression e a natural number $g(e)$ called the *Gödel number* of e. Assume that every number is the Gödel number of some expression (this is convenent but not necessary). Let e_n be the expression whose Gödel number is n. Thus, $g(e_n) = n$.

By the *diagonalization* of e_n, we will mean the expression $e_n(n)$. If e_n is a predicate, then its diagonalization is, of course, a sentence; this sentence is true iff the predicate e_n is satisfied by its own Gödel number n.

For any n, we let $d(n)$ be the *Gödel number* of $e_n(n)$. The function $d(x)$ is the diagonal function of the system. While the Gödel number of e_n is n, the Gödel number of $e_n(n)$ is not n.

	Sentences	Their Gödel numbers
Provable	**P**	P
Refutable	**R**	R
True	**T**	T

By the term *number-set*, we mean a set of natural numbers. For any number set A, by its *complement* $\overline{A}$ we mean the complement of relative to the set of natural numbers.

$\overline{A} = \mathbb{N} - A = \{n | n \in \mathbb{N} \text{ and } n \notin A\}$

For any number set A, by A^* we mean the set of all numbers n such that $d(n) \in A$ i.e., $n \in A^* \leftrightarrow d(n) \in A$. Note that $A^* = d^{-1}(A)$.

An abstract form of Gödel's theorem

Let P be the set of Gödel numbers of all the provable sentences **P**.

Theorem 19.2 (GT) - *After Gödel with shades of Tarski. If the set $\overline{P^*}$ is expressible in $\mathcal{L}$ and $\mathcal{L}$ is correct, then there is a true sentence in $\mathcal{L}$ not provable in $\mathcal{L}$.*

Proof *Suppose $\mathcal{L}$ is correct and $\overline{P^*}$ is expressible in $\mathcal{L}$. Let* H *be a predicate that expresses $\overline{P^*}$ in $\mathcal{L}$, let* h *be the Gödel number of* H *and let* G *be the diagonalization of* H *(i.e. the sentence* H(h)*). The plan is to show that* G *is true but not provable in* L. *...*

Gödel sentences

A sentence En is called a Gödel sentence for a number set A if either En is true and its Gödel number n lies in A, or En is false and its Gödel number lies outside A. Thus, En is a Gödel sentence for A iff the following holds:

$$En \in T \leftrightarrow n \in A.$$

Abstract form of Tarski's theorem

Let T be the set of Gödel numbers of the true sentences **T** of $\mathcal{L}$. Then the following theorem holds.

Theorem 19.3 (T) *(After Tarski)*

1. *The set $\overline{T^*}$ is not nameable in $\mathcal{L}$*
2. *If condition G1 holds, then $\overline{T}$ is not nameable in $\mathcal{L}$*
3. *If conditions G1 and G2 both hold, then the set* T *is not nameable in* L.

Undecidable Sentences of $\mathcal{L}$

Consistent $\mathcal{L}$ is *consistent* if no sentence is both provable and refutable in $\mathcal{L}$ (i.e., P$\cap$R=$\emptyset$) and *inconsistent* otherwise.

Decidable A sentence X is called *decidable* in $\mathcal{L}$ if it is either provable or refutable in $\mathcal{L}$ (i.e., $X \in P$ or $X \in R$) and *undecidable* otherwise.

Complete The system $\mathcal{L}$ is called *complete* if every sentence is decidable in $\mathcal{L}$ and *incomplete* if some sentence is undecidable.

Note: If $\mathcal{L}$ is correct, then it is automatically consistent as P and R are disjoint if $\mathcal{L}$ is correct.

Theorem 19.4 *If $\mathcal{L}$ is correct and if the set $\overline{P^*}$ is expressible in $\mathcal{L}$, then $\mathcal{L}$ is incomplete.*

Reference

19.2 Abstract Provability System

The following is adapted from Smullyan, see the reference at the end of this section.

Definition 19.4 Abstract Provability System
The quintuple, $\boldsymbol{M} = (S, \mathbf{f}, \supset, P, \mathbf{B})$, is a abstract provability system *where*

- *S is a set of sentences or propositions defined as follows:*
  - *$\mathbf{f}$ is a distinguished element of S called* falsehood.
  - *$\supset$ is a binary operation on elements of S such that if X, $Y \in S$, then $X \supset Y \in S$.*

  Note: S may contain symbols other than those composed of $\mathbf{f}$ and $\supset$. In particular, parentheses, brackets, and braces may be used for grouping and other symbols may be used as atomic propositions.
- *P is a subset of S whose elements are called* provable *elements of $\boldsymbol{M}$.*
- *$\mathbf{B}$ is a mapping that assigns to every element X of S an element $\mathbf{B}X$ of S i.e., $\mathbf{B} : S \to S$ and $\mathbf{B}X \in S$.*

Definition 19.5 Valuation Set
A subset V *of* S *is a* valuation set *if*

1. $\mathbf{f} \notin \mathrm{V}$
2. *For any X, $Y \in \mathrm{S}$, $X \supset Y \in \mathrm{V}$ iff either $X \notin \mathrm{V}$ or $Y \in \mathrm{V}$*
3. *X in* S *is called a* tautology *if it belongs to every valuation set.*

Exercise 19.1 *Show that there is at least one non-empty valuation set.*

Exercise 19.2 *If a valuation set contains an element, does it contain more than one element?*

Exercise 19.3 *Show that there is at least one tautology.*

Definition 19.6 Truth Set
A subset T *of* S *is called a* truth set

- *if* T *is a valuation set and*
- *if for every sentence X, the sentence $\mathbf{B}X$ is in* T *iff X is provable in* ***M*** *i.e., $X \in$ P.*

Exercise 19.4 *Show that there is a non-empty truth set.*

Abbreviations:

- $\neg X$ is $X \supset \mathbf{f}$
- $X \wedge Y$ is $\neg(X \supset \neg Y)$
- $X \vee Y$ is $\neg X \supset Y$
- $X \equiv Y$ is $(X \supset Y) \wedge (Y \supset X)$

Exercise 19.5 *Let "provable in* **M***" be defined as follows:*

> *All sentences of the following forms are provable in* **M***:*
>
> 1. $A \supset (B \supset A)$
> 2. $[A \supset (B \supset C)] \supset [(A \supset B) \supset (A \supset C)]\}$
> 3. $\neg\neg A \supset A$
>
> *and if formulas of the form A and $A \supset B$ are provable in* **M** *then so is the formula B.*

Show that all tautologies are provable in **M**.

Definition 19.7 M is of type 1
if the set of provable elements contains all tautologies and is closed under modus ponens (if X and $X \supset Y$ are both provable, then so is Y).

M is *normal* if for every provable X, the sentence **B**X is also provable.

M is *stable* if **B**X is provable, then X is also provable.

M is *consistent* if **f** is not provable.

consis Let *consis* be the sentence $\neg\mathbf{Bf}$.

Rosser mapping A mapping Q from sentences to sentences will be called a *Rosser mapping* if for every sentence X, if X is provable, then so is QX, and if X is provable, then so is $\neg QX$.

M is of *type 4* if for any sentences X and Y, the following conditions hold.

1. If X is provable, then so is **B**X (**M** is normal).
2. $\mathbf{B}(X \supset Y) \supset (\mathbf{B}X \supset \mathbf{B}Y)$ is provable in **M**.
3. $\mathbf{B}X \supset \mathbf{BB}X$ is provable in ***M***.

Theorem 19.5 *(After Tarski-Gödel) Suppose there exists a truth set T for* **M** *such that every provable element is in T, and suppose X is an element such that* $X \equiv \neg\mathbf{B}X$ *i is in T. Then nether X nor* $\neg X$ *is provable in* **M** *(yet* $X \in T$*).*

Theorem 19.6 *(After Gödel) Suppose* **M** *is a normal system of type 1 and G is a sentence such that* $G \equiv \neg\mathbf{B}G$ *is provable in* **M.** *Then*

1. *If G is provable in* **M***, then* ***M*** *is inconsistent.*
2. *If* $\neg G$ *is provable in* **M***, then* **M** *is either inconsistent or unstable.*

Theorem 19.7 *(After Rosser) Suppose* **M** *is a system of type 1 and* Q *is a Rosser mapping for* **M***. Then for any sentence X, if* $X \equiv \neg QX$ *is provable in* **M** *and* **M** *is consistent, then neither X nor* $\neg X$ *is provable in* **M.**

Theorem 19.8 *(After Gödel's Second Theorem) Suppose* **M** *is of type 4, and there is a sentence G such that* $G \equiv \neg\mathbf{B}G$ *is provable in* ***M***. *Then if* **M** *is consistent, the sentence consis (i.e., the sentence* $\neg\mathbf{Bf}$*) is not provable in* ***M***.

Theorem 19.9 *(After Löb) Suppose* **M** *is of type 4,* $\mathbf{B}X \supset X$ *is provable in* ***M,*** *and there is a sentence Y such that* $Y \equiv (\mathbf{B}Y \supset X)$ *is provable in* ***M***. *Then X is provable in* ***M***.

References

Smullyan, Raymond Gödel's Incompleteness Theorems. Oxford Logic Guides. 19. Oxford University Press.

Chapter 20

The General Setting for Decidability

A mathematical machine is an object whose behavior is defined by an *effective* finite description. As each description is finite, there are at most a countably infinite number of machines. The Halting Problem for a machine is whether or not the machine will halt for a given input. Halting problems for machines are generalizations of results that are found in recursion theory and they are related to the incompleteness theorems of logic. Regardless of the form of their description or the type of their input, Gödel numbering may be used to construct a Gödel number for each machine and a Gödel number for its input. Gödel numbering permits us to turn questions about machines and their behavior into questions about relationships between numbers.

20.1 Mathematical Machines

Let $M_0, M_1, \ldots$ be a countable sequence of mathematical machines which operate on the natural numbers $\mathbb{N}$. Each machine may be viewed as defining a set of numbers and its function is to determine which numbers are in the set and which are not in the set. Given a number and a machine, one of three things happens.

- The machine eventually halts and signifies that the number is in the set. It is said to *accept* the number.
- The machine eventually halts and signifies that the number is not in the set. It is said to *reject* the number.
- The machine does not halt as it is unable to determine whether the number is in the set.

A machine M is said to *halt at x* if given the number x, it eventually halts and signifies either that the number is in the set or that the number is not in the set.

A machine is said to be *total* if it halts on all numbers.

Two machines, M_i and M_j *behave the same way* for a number x if they both halt accepting the number, halt rejecting the number, or do not halt for that number. Two machines are said to be *similar* if for every number, they behave the same way.

Let d be a 1-to-1 function from $\mathbb{N} \times \mathbb{N}$ to $\mathbb{N}$. The function d may be viewed as defining a table of values, its arguments are the coordinates of the entries in the table. Its primary use will be related to the diagonal entries in the table $d(x, x)$. The actual function chosen is not important. It could be $2^x 3^y$, x 1's followed by y 0's, the diagonal traversal of a two dimensional table but not $x+y$ as $x + y = y + x$ violates the 1-to-1 condition.

(as done in Chapter)

For each machine M_i let M_i^d called the *diagonalizer of* M_i behave toward any number x as M_i behaves toward $d(x, x)$ that is, $M_i d(x, x)$ and $M_i^d x$ behave the same way.

Let machine M_i^{-1} accept those numbers that M_i rejects and reject those numbers that M_i accepts and does not halt on those numbers that M_i does not halt.

Let machine U, called the *universal machine*, be the machine that behaves toward $d(x, y)$ as the machine M_x behaves towards y.

Given two machines M and N, it is said that M *knows as much as* N if M accepts all numbers accepted by N and rejects all numbers rejected by N. M is not necessarily similar to N. It may accept some numbers for which N fails to halt.

A machine M is called *omniscient* if it knows at least as much as the universal machine U and is total.

An even more abstract view of mathematical machines is to view each machine as defining a partition of $\mathbb{N}$. In this approach a machine is an ordered pair of sets, (α, β). The set α is the set of numbers accepted by the machine; the set β is the set of numbers rejected by the machine; the set $\mathbb{N} - \alpha \cup \beta$ is the set of numbers for which the machine does not halt.

20.2 Exercises

1. Is the universal machine total i.e, is there is a number for which the universal machine U does not halt?

2. Problem 2

3. Problem 3

Let $\mathcal{E}$ be a countable set of sets of natural numbers and d be a 1-to-1 function $d : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ such that the following hold:

1. For any set A in $\mathcal{E}$ the set A^d of all n such that $d(n, n) \in A$ is also in $\mathcal{E}$.
2. There exists an enumeration $(\alpha_0, \beta_0), (\alpha_1, \beta_1), ...$ of all *disjoint* pairs of elements of $\mathcal{E}$ such that the set U_0 of all numbers $d(x, y)$ such that $y \in \alpha_x$ is in $\mathcal{E}$, and so is the set U_1 of all numbers $d(x, y)$ such that $y \in \beta_x$.

The pair (α, β) are called inseparable if there is no pair (α_i, β_i) such that $A \subseteq \alpha_i$ and $B \subseteq \beta_i$ and β_i is the complement of α_i.
$M_0, M_1, M_2, ...$ are mathematical machines where each machine M_i is of the form (α_i, β_i).

M is said to *accept* x if $x \in \alpha$
M is said to *reject* x if $x \in \beta$
M is said to be *silenced by* x if $x \notin \alpha \cup \beta$

$M^{-1} = (A, B)^{-1} = (B, A)$
$M^d = (A, B)^d = (A^d, B^d)$
$U = (U_0, U_1)$

CAUTION: this figure is incomplete...
$M = (\alpha, \beta)$ where $\alpha, \beta \subset \mathbb{N}$, $\alpha \cap \beta = \emptyset$.

Figure 20.1: Definitions

4. Problem 4
5. Problem 5
6. This question has two parts:
   (a) Show how to construct (via a set of rules) a machine C that behaves towards any number x as M_x behaves towards x.
   (b) Show how to construct (via a set of rules) a machine D such that for any number x, D accepts x iff M_x rejects x, and rejects x iff M_x accepts x.
7. Show that for any machine M, there is at least one number n such that U and M behave the same way towards n.
8. Suppose that machine M knows as much as machine N. Prove one of the following two statements.
   (a) One of the machines must be omniscient.
   (b) None of the machines can be omniscient.
9. Prove that there is no machine that halts at those and only those numbers on which U does not halt.
10. Why must the function d be 1-to-1?

20.3 Solutions

6a

Let C be the machine U^d, where U is any universal machine. By definition, U^d behaves toward x as U behaves towards $d(x, x)$, and U behaves towards $d(x, x)$ as M_x behaves towards x. Therefore, C behaves towards any number x as M_x behaves towards x. QED.

6b

Let D be the machine C^{-1} which is $(U^d)^{-1}$. Thus, D accepts x iff M_x rejects x, and rejects x iff M_x accepts x. QED.

1

Let M be D or any machine similar to D and let h be its index., i.e., $M_h = M$. So for any number x, M_h accepts x if and only if M_x rejects x. Apply M_h to h. M_h accepts h if and only if M_h rejects h! As this is a contradiction, M_h must fail to halt for h and by 6a, U must fail to halt for $d(h, h)$. QED.

Note: If M_h is $(U^d)^{-1}$ then U does not halt for $d(h, h)$.

7

Given machine M, let M_h be any machine similar to M^d. Now, for any x, M_h behaves toward x as M behaves towards $d(x, x)$ by definition of M^d. So, M_h behaves towards h as M behaves toward $d(h, h)$. But U behaves towards $d(h, h)$ as M_h behaves towards $d(h, h)$ and thus, so U behaves towards $d(h, h)$ as M behaves towards $d(h, h)$. Let n be $d(h, h)$, and U and M behave alike towards n. QED.

Note # 1 is immediate from this if M is U^{-1} and then U and U^{-1} be have alike towards some number n, which can only be those numbers for which they do not halt, i.e., n.

8

Pick N as a universal machine U. and then let M be any machine that knows at least as much as U. From # 7, there is a number n on which M^{-1} and U behave alike. If M accepts n then M^{-1} and U reject n and M rejects n which is a contradiction. That is, the universal machine is not omniscient. If M denies n, then M^{-1} accepts n, U accepts n, which implies M accepts n and again there is a contradiction. So M must fail to halt on n and must not be omniscient. Thus, (b) is established, QED.

Note: this solves # 1 since U knows as much as U.

9

From # 7, given a machine M, and n a number on which M and U behave alike. Then they either both halt at n or fail to halt on n.

Part IV

Logic Programming

Chapter 21

Horn Clause Logic

Horn clauses are used in the programming language Prolog.

21.1 Syntax

Terms -

```
<center>

<table NOSAVE>
  <caption>Figure 1: <b>Terms</b></caption>
  <tbody>
    <tr NOSAVE>
      <td NOSAVE>Symbols

        <blockquote>
          C = { c<sub>0</sub>, c<sub>1</sub>, c<sub>2</sub>... }; the set of
          constants <br>
          X = { x<sub>0</sub>, x<sub>1</sub>, x<sub>2</sub>... }; the set of
          variables <br>
          F =  {f<sub>0</sub><sup>0</sup>, f<sub>0</sub><sup>1</sup>,
          ... , f<sub>1</sub><sup>0</sup>, f<sub>1</sub><sup>1</sup>, ... ,
          f<sub>2</sub><sup>0</sup>, f<sub>2</sub><sup>1</sup>, ... ,
          ...}  the set of function symbols</blockquote>
        Let

        <blockquote>
          c be a syntactic variable for constants,  <br>
          x be a syntactic variable for variables, and <br>
          f be a syntactic variable for function symbols.
```

```
          <p>T ::= c | x | f(T,...,T); the set of terms.</p>
        </blockquote>
      </td>
    </tr>
  </tbody>
</table>
</center>
```

Horn clause -

```
<table NOSAVE>
  <caption>Figure 2: <b>Horn Clauses</b></caption>
  <tbody>
    <tr NOSAVE>
      <td NOSAVE>Symbols and Formulas
        <ul>
          P = { p<sub>0</sub><sup>0</sup>, p<sub>0</sub><sup>1</sup>, ... ,
          p<sub>1</sub><sup>0</sup>, p<sub>1</sub><sup>1</sup>, ... ,
          p<sub>2</sub><sup>0</sup>, p<sub>2</sub><sup>1</sup>, ... , ...} a
          set of predicate symbols</ul>
        Let p be a syntactic variable for predicate symbols
        <ul>
          A = <b>true</b> | p(T,...,T) ; a set of atomic formulas </ul>

        <blockquote>
          Let a be <b><i>a</i></b> syntactic variable for an atomic formula

          <p>G ::= <b><i>a</i></b> | G/\G - the set of goals <br>
          D ::= <b><i>a</i></b> | G -&gt; <b><i>a </i></b>| /\x.D - the set
          of positive Horn clauses</p>
        </blockquote>
      </td>
    </tr>
  </tbody>
</table>
```

Any formula of classical first-order logic can be translated to clausal form.

- Put the formula into negation normal form.
- Skolemize (replace existential variables with Skolem constants or Skolem functions of universal variables (from the outside inward). Replace
  - $\exists x.P(x)$ with $P(c)$ where c is new.
  - $\forall x....\exists y.P(y)$ with $\forall x....P(f_c(c_k))$ where f_c and c_k are new.

- Remove the quatifiers.
- Put the formula into conjuctive normal form.

Replace $C_1 \wedge ... \wedge C_n$ with the set $\{C_1, ..., C_n\}$. Each conjunct is of the form: $\neg A_1 \vee ... \vee \neg A_m \vee B_1 \vee ... \vee B_n$ which is equivalent to: $A_1 \wedge ... \wedge A_m \rightarrow B_1 \vee ... \vee B_n$

- If $m = 0$ and $n = 1$ then we have a Prolog fact.
- If $m > 0$ and $n = 1$ then we have a Prolog rule.
- If $m > 0$ and $n = 0$ then we have a Prolog query.

If n always is 1 then the logic is called Horn Clause Logic which is equivalent in computational power to the Universal Turing Machine.

Finally, replace each conjunct $A_1 \wedge ... \wedge A_m \rightarrow B_1 \vee ... \vee B_n$ with $\{A_1 \wedge ... \wedge A_m \rightarrow B_1, A_1 \wedge ... \wedge A_m \rightarrow B_2, ..., A_1 \wedge ... \wedge A_m \rightarrow B_n\}$

21.2 Sequents

From the point of view of sequents,

Sequent Axiom and Inference Rules for Horn Clause Logic

Axiom	$[U, X \vdash V, X]$		Initial sequent (leaf node)
Rules	[*set of anticedent formulas* $\vdash$ *set of succedent formulas*]		
Rule B	$\frac{[U \vdash V, \alpha]}{[U \vdash V, \alpha_1], [U \vdash V, \alpha_2]}$	$\frac{[U, \beta \vdash V]}{[U, \beta_1 \vdash V], [U, \beta_2 \vdash V]}$	MGU

An implementation is `code/hornseq` available.

21.3 Resolution

Resolution is an inference rule which requires formulas to be in clausal normal form.

Unit resolution	$\frac{A \vee B, \neg B}{A}$
Resolution	$\frac{A \vee B, \neg B v C}{A \vee C}$
Resolution	$\frac{P_1 \vee P_2 \vee ... \vee P_m, \neg P_1 \vee Q_2 \vee ... \vee Q_n}{P_2 \vee ... \vee P_m \vee Q_2 \vee ... \vee Q_n}$
Horn clause	$\frac{\neg P_1 \vee \neg P_2 \vee ... \vee \neg P_m, \neg Q_1 \wedge Q_2 \wedge ... \wedge Q_n \rightarrow P_1}{\neg P_2 \vee ... \vee \neg P_m \vee \neg Q_1 \vee ... \vee \neg Q_n}$

Chapter 22

Logic Programming

22.1 Unification

MGU - most general unifier

Occurs check

22.2 Closed World Assumption

When the semantic mapping between the language and the domain is incomplete or even missing, it may not be possible to determine whether a sentence is true or not. The closed world assumption is used provide a default solution in the absence of a better solution.

> *Closed world assumption:* if you cannot prove P or ¬P from a knowledge base KB, add ¬P to the knowledge base KB.

There are at least two situations where the closed world assumption is used. The first is where it is assumed that a knowledge base contains all relevant facts. This is common in corporate databases. That is, the information it contains is assumed to be *complete*. The second situation is where it is known that the knowledge base is incomplete (does not have enough information to produce an answer to a question) and a decision must be made without complete information - a situation familiar to most people. The closed world assumption is designed to solve a reasoning problems in both of these situations. The idea is that if you cannot prove P or not P, assume it is false. This is the usual semantics of relational databases and is employed by programs written in the programming language PROLOG. The closed world assumption is designed to finesse but not solve these problems and is adopted in default of a better solution.

The closed-world assumption simply declares that all relevant facts are stored in the database, so that any statement that is true about the actual world can be deduced from facts in the system. This assumption is useful in these situations, but it is untenable for mathematics or the scientific world. Scientific theories are, of course, rarely complete and in fact, it is their incompleteness that suggests areas for further research. The further research is designed to enlarge the knowledge base and, of course, test the accuracy of the theories.

Part V

Supplementary Material

Chapter 23

Truth Tables

The following truth tables use two values, 0 to represent false and 1 to represent true.

Negation

A	$\neg A$
0	1
1	0

Disjunction

A	B	$A \vee B$
0	0	0
0	1	1
1	0	1
1	1	1

Conjunction

A	B	$A \wedge B$
0	0	0
0	1	0
1	0	0
1	1	1

Implication

A	B	$A \rightarrow B$
0	0	1
0	1	1
1	0	0
1	1	1

Biconditional

A	B	$A \vee B$
0	0	1
0	1	0
1	0	0
1	1	1

XOR

A	B	$A\text{xor}B$
0	0	0
0	1	1
1	0	1
1	1	0

NOR

A	B	$A\text{nor}B$
0	0	1
0	1	0
1	0	0
1	1	0

NAND

A	B	$A\text{nand}B$
0	0	1
0	1	1
1	0	1
1	1	0

23.1 Exercises

1. Show that every truth function is generated by a statement form involving the connectives
   (a) $\neg, \wedge, \vee$
   (b) $\wedge, \neg$
   (c) $\vee, \neg$
   (d) $\rightarrow, \neg$
   (e) nor
   (f) nand
2. Show that the NOR and NAND connectives are the only binary connectives adequate for the construction of all truth functions.
3. Show that each of the following pairs of connectives are not adequate to express all truth functions

   (a) $\rightarrow, \vee$
   (b) $\neg, \leftrightarrow$

4. Construct three valued truth tables, undefined, false, and true.
5. Construct three valued truth tables, false, intermediate, and true.

Chapter 24

Substitution

24.1 Free and Bound Variables

If A is a formula and x is a variable but not a variable in A then so are: $\forall x.A$ and $\exists x.A$.

If $A(x)$ is formed from A by replacing any number of occurrences of some constant c with x, then the variable x is said to be *free* in $A(x)$ but is said to be *bound* in $\exists x.A(x)$ and $\exists x.A(x)$.

- x is *free* in $P^i_j(t_1, ..., t_j)$ iff x is identical with one of $t_1, ..., t_j$ where the t_i are terms.
- x is *free* in $\neg A$ iff x is free in A.
- x is *free* in $A \to B$ iff x is *free* in A or x is *free* in B.
- x is *not free* in $\forall x.A$ and is said to be *bound*.
- y is *free* in $\forall x.A$ iff y is free in A.

24.2 Substitution

Substitution in logic is *textual* substitution. It is the inspiration for passing parameters by *name* as found in the programming language Algol-60.

Substitute e for x in F (replace x with e in F)

Format	*Comments*
$\vert^e_x F$	A prefix form
$[F]^e_x$	Combined with grouping
$F\vert^e_x$	A postfix form
$F[e/x]$	Inspired by cancellation in fractions
$F[x := e], F[x \leftarrow e], F[e \rightarrow x]$	Inspired by the assignment operation
$F[x : e]$	Inspired by definition

Simultaneous substitution may be indicated by generalizing the notations.

As an example of substitution rules, the following rules illustrate substitution in logical formulas. The substitution $[e]^x_t$ (substitute x for t in e) is defined by the following rules.

Assume x, y, and t are distinct.

$[t]^x_t = x$
$[t_i]^x_t = t_i$ where $t \neq t_i$
$[P^0_i]^x_t = P^0_i$
$[F^0_i]^x_t = F^0_i$
$[P^i_j t_1 ... t_i]^x_t = P^i_j [t_1]^x_t ... [t_i]^x_t$
$[F^i_j t_1 ... t_i]^x_t = F^i_j [t_1]^x_t ... [t_i]^x_t$
...
$[A \rightarrow B]^x_t = [A]^x_t \rightarrow [B]^x_t$
...
$[\forall y A]^x_t = \forall y [A]^x_t$ where $x \neq y$
$[\forall x A]^x_t = \forall x A$
$[\exists y A]^x_t = \exists y [A]^x_t$ where $x \neq y$
$[\exists x A]^x_t = \exists x A$

Prefix notion for substition permits a reduction in parentheses.

Chapter 25

Normal Forms and Skolem Functions

25.1 Normal Forms

Normal forms are based on the expressing formulas in terms of negation, conjunction, disjunction, and the quantifiers, $\{\neg, \wedge, \vee, \forall x, \exists x\}$. An implementation is available (`code/nfs`). See `Horn.html` for Horn clause logic.

Negation normal form - NNF

In the negation normal form, negations are attached to atomic formulas. The procedure to convert a formula to negation normal form is to recursively replace formulas appearing on the left with formulas appearing on the right.

$$
\begin{array}{lll}
A \rightarrow B & \equiv & \neg A \vee B \\
A \leftrightarrow B & \equiv & A \wedge B \vee \neg A \wedge \neg B \\
\neg\neg F & \equiv & F \\
\neg(A \vee B) & \equiv & \neg A \wedge \neg B \\
\neg(A \wedge B) & \equiv & \neg A \vee \neg B \\
\neg(A \rightarrow B) & \equiv & A \wedge \neg B \\
\neg(A \leftrightarrow B) & \equiv & \neg A \wedge B \vee A \wedge \neg B \\
\neg\forall x.F & \equiv & \exists x.\neg F \\
\neg\exists x.F & \equiv & \forall x.\neg F \\
\neg\Box F & \equiv & \Diamond\neg F \\
\neg\Diamond F & \equiv & \Box\neg F
\end{array}
$$

To facilitate automated theorem proving, during the construction of the NNF, a count can be kept of the number of disjunctions in each subformula. This

information can be used to rearrange the formula so that the subformula with the fewest number of disjunctions appears on the left.

Conjunctive normal form - CNF

A formula in NNF is placed in the conjunctive normal form by recursively moving disjunctions inward and conjunctions outward using the following rewriting rules. As with the NNF, recursively replace formulas appearing on the left with formulas appearing on the right.

$$\begin{array}{lcl} A \vee (B \wedge C) & \equiv & (A \vee B) \wedge (A \vee C) \\ (A \wedge B) \vee C) & \equiv & (A \vee C) \wedge (B \vee C) \end{array}$$

Disjunctive normal form - DNF

A formula in NNF is placed in the disjunctive normal form by recursively moving conjunctions inward and disjunctions outward using the following rewriting rules. As with the NNF and CNF, recursively replace formulas appearing on the left with formulas appearing on the right.

$$\begin{array}{lcl} A \wedge (B \vee C) & \equiv & (A \wedge B) \vee (A \wedge C) \\ (A \vee B) \wedge C & \equiv & (A \wedge C) \vee (B \wedge C) \end{array}$$

Prenex normal form - PNF

A formula is placed in prenix normal form by recursively moving quantifiers outward so that all quantifiers appear at the beginning of the formula.

$$\begin{array}{lcl} Qx\neg\forall y.F & \equiv & Qx\exists y.\neg F \\ Qx\neg\exists y.F & \equiv & Qx\forall y.\neg F \\ Qx(\forall y.P \vee Q) & \equiv & Qx\forall z.(P(z) \vee Q) \\ Qx(P \vee \forall y.Q) & \equiv & Qx\forall z.(P \vee Q(z)) \\ Qx(\exists y.P \vee Q) & \equiv & Qx\exists z.(P(z) \vee Q) \\ Qx(P \vee \exists y.Q) & \equiv & Qx\exists z.(P \vee Q(z)) \\ Qx(\forall y.P \wedge Q) & \equiv & Qx\forall z.(P(z) \wedge Q) \\ Qx(P \wedge \forall y.Q) & \equiv & Qx\forall z.(P \wedge Q(z)) \\ Qx(\exists y.P \wedge Q) & \equiv & Qx\exists z.(P(z) \wedge Q) \\ Qx(P \wedge \exists y.Q) & \equiv & Qx\exists z.(P \wedge Q(z)) \\ Qx(\forall y.P \rightarrow Q) & \equiv & Qx\exists z.(P(z) \rightarrow Q) \\ Qx(P \rightarrow \forall y.Q) & \equiv & Qx\forall z.(P \rightarrow Q(z)) \\ Qx(\exists y.P \rightarrow Q) & \equiv & Qx\forall z.(P(z) \rightarrow Q) \\ Qx(P \rightarrow \exists y.Q) & \equiv & Qx\exists z.(P \rightarrow Q(z)) \end{array}$$

where Qx is the list of quantifiers and variables at the beginning of the formula and z does not occur in P or Q or in x.

Skolem Normal Form - SNF

A formula in PNF in which all existential quantifiers precede all universal quantifiers is said to be in *Skolem normal form*.

25.2 Skolem Functions

A *Skolem constant* is a new constant that is subtituted for a variable when eliminating an existential quantifier from a formula. In the formula, $\exists x.\forall y.F$, the choice of a value for x is *independent* of the choice of a value for y since once the choice for a value for x is made, it must hold for all choices for y. In this case, the variable x would be replaced by a Skolem constant c and the formula that results is: $\forall y.F(c)$.

When an existential quantifier is in the scope of a universal quantifier, the quantified variable must be replaced with a *Skolem function* of the universally quantified variables. While in the formula, $\forall y.\exists x.F$, the choice of a value for x is *dependent* on the choice of a value for y since the form asserts that for each y there is an appropriate value for x. In this case, the variable x would be replaced with a Skolem function of y and the formula that results is: $\forall y.F(skf(i,y))$.

In either case the choice of a value for y is independent of the choice of a value for x.

A good choice for Skolemizing a formula can shorten proofs. Some options include, replacing the existentially quantified variable with

- a unique constant,
- liberalized rule: from D we may infer D(c) providing the either
  - c is new or
  - the following 3 conditions all hold
    - c does not occur in D
    - c has not be previously introduced
    - no parameter previously introduced by the rule occurs in D
- a unique function of the free variables occuring in the proof,
- the formula itself.

Skolemization can be done once when a formula is placed into the NNF or whenever existential quantifiers are encountered during a proof.

Theorem 25.1 *For every formula F in language L, there is a universal formula F' in language L' with function symbols that is statisfiable iff F is satisfiable.*

Proof: *Assume the formula is in Prenex normal form. The idea is to introduce a new function symbol, f, for each existentially quantified variable, x, which takes as arguments the universally quantified variables preceding x.*

Clausal normal form

The clausal normal form is used in logic programming and many theorem proving systems. The procedure to put a formula into clausal form destroys the structure of the formula and often causes exponential blowup in the size of the resulting formula.

The procedure begins with any formula of classical first-order logic

- Put the formula into negation normal form.
- Skolemize (replace existential variables with Skolem constants or Skolem functions of universal variables (from the outside inward). Replace
  - Replace $\exists x.P(x)$ with $P(c)$ where c is new
  - Replace $\forall x....\exists y.P(y)$ with $\forall x....P(f_c(c_k))$ where f_c and c_k are new
- Remove the quatifiers.
- Put the formula into conjuctive normal form.
- Replace $C_1 \wedge ... \wedge C_n$ with $\{C_1, ..., C_n\}$. Each conjunct is of the form: $\neg A_1 \vee ... \vee \neg A_m \vee B_1 \vee ... \vee B_n$ which is equivalent to: $A_1 \wedge ... \wedge A_m \rightarrow B_1 \vee ... \vee B_n$

  Prolog and Horn Clause logic

  - If m=0 and n=1 then we have a Prolog fact.
  - If m¿0 and n=1 then we have a Prolog rule.
  - If m¿0 and n=0 then we have a Prolog query.

  If n always is 1 then the logic is called Horn Clause Logic which is equivalent in computational power to the Universal Turing Machine.
- Finally, replace each conjunct $A_1 \wedge ... \wedge A_m \rightarrow B_1 \vee ... \vee B_n$ with $\{A_1 \wedge ... \wedge A_m \rightarrow B_1, A_1 \wedge ... \wedge A_m \rightarrow B_2, ..., A_1 \wedge ... \wedge A_m \rightarrow B_n\}$.

Chapter 26

The Search for Certainty and Meaning

Out of the discussion over the paradoxes of set theory,

- Brouwer - must reject completed infinite, must reject law of the excluded middle (and non-constructive proof), must limit oneself to finite methods (constructive methods).
- Hilbert - metamathematics should be restricted to constructive methods
- Russell - must distinguish a hierarchy of types
- Practicing mathematicians should
  - show that objects are well defined by demonstraiting the existence of the object of interest (preferably by construction).
  - be explicit when the axiom of choice (or equivalent) is used

26.1 Plato, Brouwer, Hilbert, and Tarski

The Platonic Universe

Fundamental and abstract entities exist in a platonic realm inaccessible by means of our standard perceptual capacities. Any statement about the properties of a given *finite or infinite* mathematical domain is either true or false.

Acceptance of the Platonic assumptions imply completed infinity and validity of non-constructive proof.

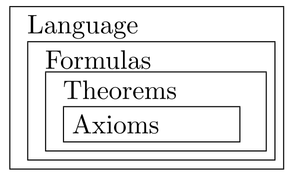

Figure 26.1: The Formal Hierarchy

Formalism (David Hilbert, Bertrand Russell)

Mathematical truth is determined by proof. All provable statements are true and all true statements are provable. Mathematics is the science of formal systems. A formal system consists of

- A language. The language is a set of strings. The strings are sequences of symbols (selected from a set of symbols) described by a grammar.
- A set of rules of inference. A rule of inference describes a relation between a set of strings and a string (an inferred string) .
- A set of axioms which are a subset of the language. The axioms form the starting point for the rational exploration of the system.

This formalist hierarchy is illustrated in Figure 26.1.

The goal of the mathematical "game" is to determine which sentences of the language are reachable from the set of axioms using the rules of inference. If the system includes a complementary operator, then both a sentence and its complement must not be reachable (Law of the excluded middle).

Given a proposition (a string in a language), one of three possibilities holds.

1. The proposition is derivable from the axioms using a finite number of applications of the rules (Provable).
2. If the system includes a complementary operator, then the complement of the proposition is derivable from the axioms (Law of the excluded middle)).
3. Neither the proposition nor its complement is derivable from the given axioms (Gödel's incompleteness theorem).

See the `../CII/MathGame.html` on Hilbert's Math Game for more details

See `../Poetry/fable.html` The Battle of the Frog and the Mouse

Correspondence theory of truth (Alfred Tarski)

Mathematical truth is defined using a satisfaction function between the syntax of a language and the mathematical structures that provide the semantics for the language. The truth of a proposition is dependent on a mapping from language to a structure. The language must correspond to some reality. In Tarski's view, the sentence X of language L **follows logically** from the sentences of the class K if and only if every model of the class $\mathbf{K}$ is a model of the sentence X.

See the `../CII/MGTarski.html` on Tarski's Math Game for more details.

Digression on Consistency

A theory is *consistent* if

- it has a model or
- both P and $\neg P$ cannot be true or
- not all propositions are true (important for systems without negation).

Intuitionism (L. E. J. Brouwer)

Mathematical truth is perceived by intuition and reported by linguistic methods. Any statement about the properties of a given *finite* mathematical domain is either true or false.

1. Objects must be constructed
2. Law of the excluded middle $(p \vee \neg p)$ is disallowed.
3. Infinite sets are disallowed

See `Intuitionistic.html`Intuitionism for more details

See `../Poetry/fable.html`The Battle of the Frog and the Mouse

Constructive Logic (Constable)

In the applied sciences (and in particular computer implementations), it is not sufficient to claim to have proven the existence of some property without being able to produce an object with that property. It is not sufficient to claim to have produced a proof of the existence of a function without providing a way to calculate it. Constructive logic is a blending of formalism and intuitionism which produces effective methods for constructing objects of interest. In constructive logic, a proof of $\exists x.P(x)$ must provide a way to construct an object c and show that $P(c)$ is true and a proof of $p \vee q$ must provide a proof of p or a proof of q, that is, it is not sufficient to prove $\neg(\neg p \wedge \neg q)$.

See also the `lambdaCalculus.html`Lambda Calculus and `turingMachine.html`Turing `http://www.ac.wwu.edu/~skywise/turingMachine.html`Machines

Chapter 27

Coherence, Correspondence, and Compromise

This chapter is about theories of information, meaning, and truth.

27.1 Coherence theory

A coherence theory of meaning states that the meaning (or truth) of any proposition consists in its coherence with some specified set of propositions. The relation between propositions and their truth conditions is coherence (consistency). The truth conditions of propositions consist in other propositions. There are two extreme positions, a statement is true if no contradictions occur when it is added to the set of propositions and a set of proposition is true if there is a proof that they are not contradictory. This is the approach is used in some forms of literary analysis and by the formalist school of mathematics and it may be called *truth by coherence*.

An example of coherence is in narratives. Three types of coherence may be identified.

- Structural coherence - the parts of the story "hang together"
- Material coherence - fits with other stories
- Characterological coherence - believability of the characters

Finite collections of statements expressed in some languages (typically propositional logics) have the finite model property which provides a means of determining if the set of statements is satisfiable (consistent). The finite model

property actually permits the construction of a structure in which the statements are satisfied. However, we know from Gödel's incompleteness theorem that in general, it is impossible to prove the consistency of a consistent set (collection) of propositions from the set (collection) of propositions.

27.2 Representational (correspondence) theory

The correspondence theory of meaning states that the meaning of any proposition consists is its correspondence with a feature of the world. The truth conditions of propositions are objective features of the world and are precise and complete. This is the approach taken by mathematicians and it may be called *truth by design*.

Truth (equivalently, meaning or validity) is a property of statements in a language determined by a correspondence with objects in structure. In the correspondence theory there are two systems, a representing system (the language) and a represented system (a relational structure), and a correspondence or relation between the two systems. The correspondence between the systems determines what is variously called truth, validity, meaning, or semantics i.e., the meaning, of a proposition consists in its correspondence with a feature of the relational structure. When the representing system is a logical language, the correspondence is called a valuation function. When the representing system is a programming language, the correspondence is called a semantic function. Valuation functions determine the validity of a statement from the validity of the parts of the statement and semantic functions determining the meaning of a program from the meaning of the parts of the program. The correspondence theory has worked well for mathematics and theoretical computer science where the truth conditions of propositions are objective features of the relational structure.

> **Aside:**This definition suggests that words have meaning but things do not. This implies that life really has no meaning. That this is not correct as may be seen when the definition is generalized by recognizing that the key element in meaning is the correspondence between two sets of objects and the kind of manipulations one may perform on objects in the first set. So, for example, a doll may be associated with a memory, a telephone with an opportunity for communication. Events may be given meaning by virtue of the associated consequences. However, it is not useful to suggest that all correspondences and relationships should constitute meaning. Meaning should be reserved for use in describing those relationships that are part of rational activities.

Formally, the correspondence theory is implemented in a formal system. The key features are illustrated in Figure 27.1.

Representing system (meta level)	*Meaning (semantic mapping, valuation function)*	*Represented system (object level)*
a, ...	$\rightarrow$	**a**, ...
Formal, abstract, proof, language	Vague, precise, valid (true), accurate	Informal, concrete, truth, structure
	Examples:	
assembly instructions a road map a mathematical theory a scientific theory	$\rightarrow$	an unassembled product a road system a mathematical structure the natural world

Figure 27.1: Representational theory of information

Suppose there is a class consisting of a teacher and three students. A language for the class can be constructed as follows. The members of the class are teacher(a), student(b), student(c), student(d). One aspect of experimental science is discovery of the properties of the phenomena under study and the construction of a simple language for those properties. The requirements phase of an engineering project also focuses on determining the properties of the problem domain and a construction of a suitable language for its description. In the maintenance mode of software projects, the mapping is an on going work of the stakeholders.

Wicked problems are a class of problems that are particularly difficult to solve because the problem changes during its solution. ... ???

Semantic relations

A semantic relation may be 1-to-1 giving an accurate representation. However, it is not always possible to construct accurate representations. For example, the word "red" is mapped to a number of actual colors and ambiguously to many others. Most mappings in scientific theories are vague. A vague mapping is 1-to-many. Bertrand Russell's `vaguerus.html`paper on vagueness is a highly readable discussion of these ideas. The following definitions are helpful in creating a language for semantic relations.

- The relation between the representing system A and the represented system B is *meaning*. The relation is
  - an*injection (1-to-1)* if each element of A maps to a unique element in B,
  - a *surjection (onto)* if for each element of B there is an element of A that maps to it,
  - *a bijection* if it is both 1-to-1 and onto,
  - *1-to-many* if some element of A maps to more than one element in B,
  - *partial* if some element of A does not map to an element of B,
  - *nondeterministic* if the mapping of some element of A to elements of B varies over time, and
  - other relationships include *speculative*, *probabilistic*, *fuzzy*, and *chaotic*.
  - *computable* if it is an injection and is a computable function,
  - *decidable* if there is an effective procedure for determining whether or not b=f(a).
  - Relations defined by rules of deduction are not, in general, decidable according to Gödel's incompleteness theorem.
- A statement in a representing system is

  - *vague* if many possible facts in the represented system could verify it and it is undecidable whether a fact verifies it,
  - *ambiguous*: if many possible facts in the represented system could verify it and it is decidable whether a fact verifies it,
  - *precise* if only one fact in the represented system would verify it ,
  - *valid* (*true*, corresponds to reality) if there is a corresponding fact in the represented system and
  - *accurate* if it is both precise and valid.

- A representing system is
  - *vague* if some statement in the representing system is vague (the semantic mapping is 1-to-many and undecidable),
  - *ambiguous* if some statement in the representing system is ambiguous (the semantic mapping is 1-to-many),
  - *precise* if each statement in the representing system is precise (the semantic mapping in an injection),
  - *valid* if each statement in the representing system is valid (the semantic mapping is verifiable),
  - *accurate* if it is both precise and valid (the semantic mapping is an injection and verifiable),
  - *incomplete* (underspecified) if some fact in the represented system does not correspond to a statement in the representing system (the semantic mapping is not a surjection),
  - *complete* if the semantic mapping is a bijection (injection and surjection), and
  - *partial* (over specified) if some statement in the representing system does not correspond to a fact in the represented system (the semantic mapping is a partial function).

The following observations are useful. A vague belief has a much better chance of being true than a precise one because there are more possible facts that would verify it. The more specific (precise) a claim, the less likely it is to be true. A precise belief is harder to be true but better worth having if it is true. "Science is trying to substitute more precise beliefs for vague ones; this makes it harder for a scientific proposition to be true than for vague beliefs but make scientific truth better worth having if it can be obtained." - B. Russell.

> **Aside:**Words have meaning by virtue of their correspondence to some action or object. Actions may have consequences. Objects and actions may have explanations in some context. A consequence of a correspondence between words and objects in a structure is that words have meaning. Thus it is correct to say that the universe has no meaning.

Representing system	*Accurate semantic mapping*	*Represented system*
Language: a, b, ... and logical symbols.	$\rightarrow$	Code: **0, 1, ...**
The semantic mapping is one-to-one.		

Figure 27.2: Secret code: Encryption and decryption

Aside:A correspondence between a and b is

- decidable if given a and b, it is decidable that I(a)=b,
- computable, if given a, I(a) is computable,
- verifiable, if

Examples

Semantic mappings may do more than just map simple objects from the representing system to the represented system. The complexity of a semantic mapping depends on the complexity of the language. A simple one-to-one mapping between letters and numbers is sufficient for a simple secret code (See Figure 27.2).

Formal languages like those employed in logic (see Figure 27.3) and mathematics are more complex and require extensions that capture the meaning of the logical connectives (not, and, or, implies, etc.).

A less language like example is that of scientific models and physical reality (Illustrated in Figure 27.4). For science the vague semantic mapping between the abstract, idealized mathematical structure of theoretical science and the represented physical structure is problematic because we do not have complete or accurate information on the represented physical structure. In spite of the problems, the resulting mathematical models in the physical sciences have proved to be surprisingly useful. In the so called "softer" sciences, the proposed semantic maps have proved to be less reliable and even controversial.

The semantic mapping between natural language text and a logical language, possibly a default logic, is complex (Illustrated in Figure 27.5). The semantic mapping must preserve the usual the lexical mappings, the grammatical mappings, and contextual meanings. For historical documents, the semantic

Representing system	*Precise semantic mapping*	*Represented system*
Propositions: a, b, ... and logical symbols.	→	Facts: **a, b, ...**
The semantic mapping is one-to-one. The valuation function v is defined in terms of the semantic mapping and preserves logical relations: v(a) is true if and only if $m(a) =$**a**; v(P and Q) is true if and only if both v(P) and v(Q) are true; etc.		

Figure 27.3: Propositional Logic

Representing system	*Vague semantic mapping*	*Represented system*
Abstracted and idealized mathematical model.	→	Physical system
The semantic mapping involves the use of approximations such as simplifying assumptions and generalizations to produce a computationally tractable mathematical model.		

Figure 27.4: Science: mathematical models and physical reality

Representing system	*Semantic mapping*	*Represented system*
Default logic	$\rightarrow$	Natural language text
The semantic mapping satisfies lexical, grammatical,and contextual, and other requirements.		

Figure 27.5: Natural language text

mapping must conform to the larger historical context as well. For sacred documents, there is often violent disagreement over the proper semantic mapping and the disagreement often leads to the creation of different schools of thought, sects, and denominations.

The difficulty of constructing semantics maps and the difficulty in objectively analyzing the represented system have lead some to use the coherence theory of meaning where the representing system is constructed by some intuitive sense. More formal approaches use `http://cs.wwc.edu/~aabyan/Logic/MultiValued.html`multivalued logics and default logics to provide a vague semantic map and the option of revising the logic as additional and more complete information is acquired about the represented system.

References and Notes

Besnard, P. *An Introduction to Default Logic* Springer-Verlag 1989. ISBN 3540515666

Russell, Bertrand Vagueness in *The Australasian Journal of Psychology and Philosophy,* 1 (June 1923): 84–92

Rorty, Richard Rorty argues against the correspondence theory if it is used as an absolute theory. No reference available.

Stanford Encyclopedia of Philosophy Tarski's Truth Definitions

27.3 Social consensus theory

The social consensus theory of meaning states that the meaning of any proposition consists in its designation as meaningful by some social group. The relation between propositions and their truth conditions is an ongoing achievement (work) of some social group. The truth conditions are situated, local, contingent, embodied, vague, and open. This is the approach taken by social institutions and it may be called *truth by social agreement*. The material in this section has been extracted from various papers by Joseph A. Goguen.

Definition 27.1 *An item of information is an interpretation of a configuration of signs for which members of some social group are accountable.*

Elaboration: *Meaning* is an ongoing achievement of some social group; it takes work to interpret configurations of signs, and this work necessarily occurs in some particular context, including a particular time, place, and group. The meaning of an item of information consists of the relations of accountability that are attached to it in that context, and the narratives in which it is embedded.

The consequences of information tied to a particular, concrete situation and a particular social group are:

1. *Situated.* Information can only be understood in relation to the particular, concrete situation in which it actually occurs.
2. *Local.* Interpretations are constructed in some particular context, including a particular time, place, and group.
3. *Emergent.* Information can only be understood through the ongoing interactions among members of a group. These interactions involve negotiation and compromise.
4. *Contingent.* The interpretation of information depends on the current situation which may include the current interpretation of prior events.
5. *Embodied.* Information is tied to bodies in particular physical situations, so that the particular way that bodies are embedded in as situation may be essential to some interpretations.
6. *Vague.* Information is only elaborated to the degree that it is useful to do so; the rest is left grounded in tacit knowledge.
7. *Open.* Information is open to revision in the light of further analysis and further events.

Groups, values, and information are *coemergent*, in the sense that each produces and sustains the other; values exist because they are shared and communicated by groups; and information exists because groups share values in a dynamic world.

Representing system (meta world) (a configuration of signs)	*mapping* → 1 to n	*Represented system (object world)*
Formal abstract language	An ongoing achievement (work) of some social group. The mapping is situated, local, contingent, embodied, vague, open	Particular context including time, place, group, relations of accountability, and narratives (text, video, sound recordings)
	Examples	
Church doctrines	A particular church body	Scriptures and a group
Software requirements	Requirements engineers and stakeholders	Needs and goals - the high-level objectives of the system, and the users/stakeholders.

Figure 27.6: Social theory of information

Commentary

Figure 27.6 puts the social theory of information in the format used for the `Correspondence.html` correspondence theory of truth Since the mapping between the representing system and the represented system is dynamic in a number of dimensions, the mapping may be described as

- nondeterministic,
- an optimization with respect to competing values/perspectives of the members of the group, or
- an infinite valued, non-monotonic, temporal logic as opposed to classical two valued Aristotelian logic.

Identification

The identifying characteristics of a social ethical information system are the following:

- A dynamic domain (the object world).
- A configuration of signs (a dynamic meta world).
- A social group embedded in the dynamic domain which is responsible for maintaining a mapping from the meta world to the object world.
- The mapping often
  - involves contradictory positions that are separated in time and
  - myths rooted in a historical events that persist in the face of objective counter evidence.

References

Goguen, Joseph (1997) Towards a Social, Ethical Theory of Information, by Joseph Goguen, in Social Science Research, Technical Systems and Cooperative Work, edited by Geoffrey Bowker, Les Gasser, Leigh Star and William Turner (Erlbaum, 1997) pages 27-56.

Goguen, Joseph (1994) Requirements Engineering as the Reconciliation of Technical and Social Issues, in Requirements Engineering: Social and Technical Issues, edited with Marina Jirotka (Academic Press, 1994) pages 165-199.

Goguen, Joseph and Linde, Charlotte (1993) Techniques for Requirements Elicitation, by Joseph Goguen and Charlotte Linde, in Proceedings, Requirements Engineering '93, edited by Stephen Fickas and Anthony Finkelstein, IEEE Computer Society, 1993, pages 152-164.

27.4 Other theories

A constructive/computation theory of meaning states that the meaning of a statement depends on whether it is computable. This is the approach taken by the intuitionistic school of mathematics and it may be called *truth by construction.*

A verification theory of meaning states that the meaning of a statement depends on whether it is verifiable. This is the approach taken by science and it may be called *truth by experiment.*

In the case of the revealed literature used in various religions, the notion of truth is *truth by divine revelation.*

Chapter 28

A Taxonomy of Theories

> This work makes several original contributions. The first is a descriptive rubric based on three areas of philosophy, ontology, epistemology, and axiology. It is employed to describe theories. The second is a hierarchical taxonomy for theories and a taxonomy for empirical theories. This taxonomy clarifies the relationships between theories in mathematics and science and theories in other disciplines. The third (which is developed in another paper [1]) is the definition and use of basic metaphorical reasoning primitives as fundamental epistemological methods and as heuristics used in the evolution of theories. ... non-rational reasoning methods ... And last, is the use of the ISO Standard 9126 as a taxonomic framework for the values appropriate for theories.

28.1 Overview

Theories occur in many disciplines. In theology they are called doctrines. In literature they are called literary interpretations (usually constructed using a particular literary theory). In engineering they are called designs or models. In the sciences they are theories or models. In software engineering there are many different kinds of theories, requirements, specifications, designs, and implementations are all types of theories. Software in execution is a product but its code is a theory that can be used to predict the behavior of its execution.

This work provides a uniform framework for understanding the varieties of theories that develop in different disciplines. The initial motivation for this work is the need to utilize theories from several different disciplines in the various phases of the software life-cycle. [1]

[1] *Theories in mathematics and logic* are precise descriptions of the properties of completely describable, universe. *Scientific theories* are vague (within a margin of error) descriptions

This work makes several original contributions. The first is a descriptive rubric based on three areas of philosophy, ontology, epistemology, and axiology. It is employed to describe theories. The second is a hierarchical taxonomy for theories and a taxonomy for empirical theories. This taxonomy clarifies the relationships between theories in mathematics and science and theories in other disciplines. The third (which is developed in another paper [1]) is the definition and use of basic metaphorical reasoning primitives as fundamental epistemological methods and as heuristics used in the evolution of theories. ... non-rational reasoning methods ... And last, is the use of the ISO Standard 9126 as a taxonomic framework for the values appropriate for theories.

Methodological considerations are covered in Sections 28.2-28.4. Section 28.2 is an ontological description of theories. Subsection 28.2.1 consists of the standard description of formal systems and may be skipped on first reading. Subsection 28.2.3 contains formal descriptions of various types of theories. Subsection 28.2.2 consists of the standard descriptions of complexity. Section 28.3 is a description of the epistemological methods used to construct theories and a hierarchical taxonomy of theories. Section 28.4 is a description of the values that lead to different types of theories. Section 28.5 is a summary, conclusions, and suggestions for further research. Section 28.5 is a reading list and references.

28.2 Ontology for Theories

Ontology is the theory of objects and their ties. It provides criteria for distinguishing various types of objects (concrete and abstract, existent and nonexistent, real and ideal, independent and dependent) and their ties (relations, dependences and predication).

28.2.1 Syntax, Semantics, Definitions, and Consequences

We begin with an informal definition of a theory.

Definition 28.1 *A* theory *is a compact effective description (Hilbert) of a domain of interest (Tarski) and effective rules of inference for reasoning about the domain of interest.*

---

of the invariant (recurring) properties of a complex, dynamic universe. *Speculative or forensic scientific theories* are the application of standard scientific theories to reconstruct nonrecurring (historical) events or predict potential future events. *Social science theories* are descriptions of non-recurring properties of a dynamic universe. *Myths* are theories that are culturally significant stories or explanations which serve several purposes. Logic and mathematics study the logical relationships in relational structures with the goal of providing a effective, minimal complete description. Science studies recurring events in empirical domains with the goal of understanding unique events. In the humanities, unique events are studied with the goal of understanding recurring events.

Theory $\stackrel{\Delta}{\rightarrow}$ Relational Structure

Where $\stackrel{\Delta}{\rightarrow}$ designates a defined functional relationship.

Figure 28.1: **Semantic Function**

Figure 28.1 illustrates the functional relationship between a theory, the structure it represents and a semantic function that describes the relationship. The domain of interest may be simple or complex, static or dynamic, real or imaginary. The domain of interest is often a *relational structure* which is an intellectually manageable abstract representation of the actual domain of interest which may be an empirical domain. More formally,

Definition 28.2 Relational Structure. *A* relational structure *is a well-defined, possibly infinite set of objects, attributes, and relationships (both static and dynamic) i.e.,* $\mathcal{D} \subset \bigcup_{i \in \mathbb{N}} \wp(C^i)$

There are always infinitely many theories compatible with any finite body of data. The critical question is how to make appropriate choices from this infinite range of options.

Logical Fertility – The description of relational and empirical domains includes objects, their attributes, and static and dynamic relationships. The description is formulated to permit logical manipulation and computation. There are two types of connections

1. formal connection - a purely logical relation to other constructs; these are universal rather than particular instances.

2. epistemic connection - a link between construct and data (i.e., there must be a semantic connection between constructs and data).

Causality/Predictive Power – In dynamic systems there is a relation between states of a physical system. The constructs are chosen so as to generate causal laws to predict future states of the system i.e., calculate the behavior of the system or the controlling actions for the system.

A formal system consists of a language, a structure, a correspondence between the language and the structure and some inference rules that describe how to construct a new sentence from zero or more sentences. Axioms, sentences in the language, are used to describe the properties of the structure and the inference rules support reasoning about the structure. The language is constructed to provide a precise description of the structure and facilitate the formulation of questions and to facilitate reasoning about the objects and properties derived from the description of the structure. The relationship between the primitive sentences in the language and the elements of the structure is one-to-one.

Definition 28.3 Language. *Let Σ be a possibly countably infinite* set of symbols*, Σ^* be the set of all finite strings of symbols in Σ and $\mathcal{L} \subset \Sigma^*$ be a finitely describable* language *(a set of sentences or strings).*

Languages are usually described using a grammar with a finite number of rules. Logic languages usually contains operators for negation ($\neg$), disjunction ($\vee$), conjunction ($\wedge$), and implication ($\rightarrow$) among others.

Definition 28.4 Model. *A language $\mathcal{L}$ for a relational structure $\mathcal{D}$ includes a* semantic relation *$\sigma \subset \mathcal{L} \times \mathcal{D}$. $\mathcal{D}$ is the co-domain of σ.*

> *$\mathcal{L} \rightarrow \mathcal{D}$ designates the set of precise semantic functions and*
> *$\mathcal{L} \rightsquigarrow \mathcal{D}$ designates the set of vague semantic relations.*

$\mathcal{M}$ is the domain of the semantic relation σ and is said to model *the relational structure, $\mathcal{D}$. $\overline{\mathcal{M}}$ (the set of true statements) is the closure of $\mathcal{M}$ with respect to the logical operators, $\neg$ and $\vee$, if for every $f \in \mathcal{L}$, $f \in \overline{\mathcal{M}}$ iff*

> *$f \in \mathcal{M}$,*
> *$f = \neg A$ and $A \notin \overline{\mathcal{M}}$, or*
> *$f = A \vee B$ and $A \in \overline{\mathcal{M}}$ or $B \in \overline{\mathcal{M}}$.*

Notation: $\overline{\mathcal{M}} \models f$ iff $f \in \overline{\mathcal{M}}$; $\overline{\mathcal{M}} \models A$ iff $A \subset \overline{\mathcal{M}}$.

Definition 28.5 Theory. *A* theory *$\mathcal{T} \subset \mathcal{L}$ is a finite subset of a language. The theory is a description of some domain $\mathcal{D}$. The elements of the theory are called* axioms*.*

Definition 28.6 Inference Rule. *A set of* inference rules *$\mathcal{I} \subset \wp(\mathcal{L}) \times \mathcal{L}$ is a collection of relationships between sets of formulas and a formula. The inference rules must have a finite description of the form:*

$$\frac{\textit{Premises: } A_1, ..., A_n}{\textit{Conclusion: } B}$$

A *deduction* is a rule of inference in which the conclusion follows necessarily from the premises. An *induction* is a rule of inference in which the conclusion is supported by the premises but does not follow necessarily i.e., it is possibly false. *Scientific induction* is an evolving collection of objective inductive rules of inference for empirical phenomena. The validity of a rule of scientific induction depends on statistical measures of its reliability. *Creative induction* is a novel application of a metaphor to solve a problem.

Definition 28.7 Proof. *A* proof *is a finite directed acyclic graph (DAG) that satisfies the following:*

- *The nodes are labeled with elements of $\mathcal{L}$.*

- *The edges represent the application of inference rules.*
- *The nodes with no incoming edges are labeled with elements of $\mathcal{T}$.*
- *There is just one node with no outgoing edge. It is called a* theorem.

Usually a proof is defined as a sequence of statements each of which is an element of $\mathcal{T}$ or is derived from previous statement in the sequence by an application of an inference rule. Such sequences are topological sorts of the DAG. A proof may also be represented as a tree with the root labeled with the theorem and the leaves labeled with elements of $\mathcal{T}$.

Definition 28.8 Theorems. $\overline{\mathcal{T}}$ *(the set of* theorems) *is the closure of $\mathcal{T}$ with respect to a set of inference rules $\mathcal{I}$ if for every $f \in \mathcal{L}$, $f \in \overline{\mathcal{T}}$ iff*

$f \in \mathcal{T}$ *or*
$f = C$ *where* $H \subset \overline{\mathcal{T}}$, $(H, C) \in i$, $i \in \mathcal{I}$

Notation: $\mathcal{T} \vdash f$ *iff* $f \in \overline{\mathcal{T}}$, $\mathcal{T} \vdash A$ *iff* $A \subset \overline{\mathcal{T}}$

$\overline{\mathcal{T}}$ may be represented as a directed graph that satisfies the following:

- The nodes are labeled with the elements of $\overline{\mathcal{T}}$.
- The edges represent the application of inference rules. The directed edge may be used to represent either dependency or the direction of inference.
- If edges point in the direction of the conclusion, then the nodes with no incoming edges are the elements of $\mathcal{T}$.

For a given theory $\mathcal{T}$, $\mathcal{M} \models \mathcal{T}$ denotes that $\mathcal{M} \models a$ for each sentence a in A. $\mathcal{T} \models p$ denotes that for every model $\mathcal{M}$, $\mathcal{M} \models p$ whenever $\mathcal{M} \models \mathcal{T}$. If F is a set of sentences, $\mathcal{T} \models F$ denotes that for every A in F and for every model $\mathcal{M}$ such that $\mathcal{M} \models T$ then $\mathcal{M} \models \mathcal{A}$.

Definition 28.9 Satisfiable. *A theory $\mathcal{T}$ is satisfiable if there is a structure and semantic function $\mathcal{M} = \langle S, \sigma \rangle$ such that $\mathcal{M} \models \mathcal{T}$.*

Definition 28.10 Valid theory *A theory $\mathcal{T}$ is valid if for all appropriate structures and semantic functions $\mathcal{M} = \langle S, \sigma \rangle$, $\mathcal{M} \models \mathcal{T}$. The sentences in $\mathcal{T}$ are called* tautologies.

A formula is valid if and only if its negation is not satisfiable. A theory is valid if and only if the negation of any one of its formulas is not satisfiable.

Definition 28.11 Sound Inference Rule. *An inference rule $i \in \mathcal{I}$ is* sound *iff for every $(H, C) \in i$ whenever $H \subset \overline{\mathcal{M}}$ then $C \in \overline{\mathcal{M}}$*

The inference rule *modus ponens* is a familiar example of a sound inference rule.

$$\text{Modus Ponens: } \frac{\text{From: } A \text{ and } A \rightarrow B}{\text{Infer: } B}$$

Some inference rules can be shown to be unsound. Others are difficult to formalize or do not produce reproducible results. The following are examples of unsound inference rules.

> *Mysticism:* An inference achieved through a non-rational experience.
> *Superstition:* An inference based on the assumption of a causal link between previous random events juxtaposed in time.
> *Random Choice:* An inference determined by a random event such as a coin toss.
> *Imagination:* An inference which creates concepts and meaningful forms of what is not present.
> *Intuition:* An inference based on experience.

Definition 28.12 Consistent Theory. *A theory is consistent if it contains no contradiction i.e., it is not the case that both f and $\neg f$ are in $\overline{\mathcal{T}}$.*

Satisfiability and consistency are related. Consistent theories are satisfiable. In order to achieve this one or more of the sentences may need to be modified through a reduction in generality or even deleted from the theory.

Definition 28.13 Contradictory Theory. *A theory is contradictory if it is unsatisfiable.*

Inconsistent theories are contradictory theories.

Axioms and Theories

Definition 28.14 Independent. *Two sentences are* mutually independent *of each other if neither can be derived from the other.*

From the notion of proof, two sentences A and B are mutually independent of each other if $A \notin \overline{\mathcal{T} \cup \{B\}}$, and $B \notin \overline{\mathcal{T} \cup \{A\}}$. From the notion of validity, two sentences A and B are mutually independent of each other if there are two models. M_1 and M_2 such that $M_1 \vdash A \wedge \neg B$ and $M_2 \vdash \neg A \wedge B$. From the notion of necessity, if two distinct sentences are necessary, then they are *independent.* However, independence does not imply necessity as both sentences may not be required in a given proof.

Independence may be extended to theories.

Definition 28.15 Independent Theories. *Two theories are independent of each other if their closures have nothing in common i.e., $\mathcal{T}_1$ and $\mathcal{T}_2$ are independent if $\overline{\mathcal{T}_1} \cap \overline{\mathcal{T}_2} = \emptyset$.*

Alternately, there are two theories, $\mathcal{T}_1, \mathcal{T}_2$, and two models, M_1, M_2, such that $M_1 \models \mathcal{T}_1$ but not $M_1 \models \mathcal{T}_2$ and $M_2 \models \mathcal{T}_2$ but not $M_2 \models \mathcal{T}_1$.

We may define two theories to be dependent if they are not independent i.e., they have something in common. Having something in common ranges from just one sentence to having every sentence in common. Two extreme situations occur (1) the closure of one is a subset of the other and (2) the closures are equal. The first case is said to be logically weaker [2] , the second is equivalence.

Definition 28.16 Logically Weaker. *One theory is logically weaker than another theory if its closure is a subset of the closure of the other theory i.e., $\mathcal{T}_1$ is weaker then $\mathcal{T}_2$ if $\overline{\mathcal{T}_1} \subset \overline{\mathcal{T}_2}$.*

Theory $\mathcal{T}_2$ is logically stronger or more general then $\mathcal{T}_1$.

Definition 28.17 Parsimony. *If two theories are equivalent, the more parsimonious theory is either a proper subset of the other or contains fewer formulas i.e., $\mathcal{T}_1$ is a smaller theory if $\mathcal{T}_1 \equiv \mathcal{T}_2$ and either $\mathcal{T}_1 \subset \mathcal{T}_2$ or $|\mathcal{T}_1| < |\mathcal{T}_2|$.*

The alternative $|\mathcal{T}_1| < |\mathcal{T}_2|$ is necessary because the theories might contain completely different axiom sets yet generate the same theorems.

Definition 28.18 Equivalent Theories. *Two theories are equivalent if each may be derived from the other i.e., $\mathcal{T}_1 \equiv \mathcal{T}_2$ iff $\mathcal{T}_1 \vdash \mathcal{T}_2$ and $\mathcal{T}_2 \vdash \mathcal{T}_1$.*

Alternatively, their closures are the same i.e., $\mathcal{T}_1 \equiv \mathcal{T}_2$ iff $\overline{\mathcal{T}_1} = \overline{\mathcal{T}_2}$.

Definition 28.19 Sufficient. *A theory $\mathcal{T}$ is* sufficient *for the proof of a sentence, t,if no other axiom is required to construct a proof of t, i.e., $\mathcal{T} \vdash t$.*

Definition 28.20 Necessary. *A sentence is said to* necessary *if its use (in a proof) is required.*

Logical Consequences

The following are well known properties of sound theories i.e., theories formulated in formal logic with sound inference rules.

Definition 28.21 Sound Theory. *If a formula is provable, it is also true i.e., for all formulas f in $\mathcal{L}$, if $\mathcal{M} \models \mathcal{T}$ and $\mathcal{T} \vdash f$, then $\mathcal{M} \models f$.*

Theorem 28.1 *Propositional and first-order logic are sound and valid theories.*

[2] One proposition is logically weaker than a second proposition if the second proposition implies the first.

Lemma 28.1 *The following are equivalent:*

1. $\mathcal{T}$ *is a contradictory theory*
2. $\overline{\mathcal{T}} = \mathcal{L}$
3. $\mathcal{T} \vdash f$ *and* $\mathcal{T} \vdash \neg f$

Lemma 28.2 *Consistency and satisfiability are equivalent.*

Definition 28.22 Decidable Theories. *In decidable theories, all statements are either provable or disprovable.*

$$\textit{Decidable: For all } P \in \mathcal{L}, \textit{ either } \mathcal{T} \vdash P \textit{ or } \mathcal{T} \vdash \neg P$$

e.g. Boolean Algebra.

Definition 28.23 Complete Theories. *In complete theories, all true statements are provable.*

$$\textit{Complete: For all } P \in \mathcal{L}, \textit{ if } \mathcal{M} \models P \textit{ and } \mathcal{M} \models \mathcal{T} \textit{ then } \mathcal{T} \vdash P$$

e.g. First-order Logic

Definition 28.24 Incomplete Theories. *In incomplete theories, some true statements are unprovable.*

$$\textit{Incomplete: For some } P \in \mathcal{L},\ \mathcal{M} \models P,\ \mathcal{M} \models \mathcal{T}, \textit{ and not } \mathcal{T} \vdash P$$

e.g. Arithmetic

Theorem 28.2 Gödel's Completeness Theorem. *First-order logic is complete but not decidable.*

Theorem 28.3 Gödel's Incompleteness Theorem. *First-order arithmetic is incomplete.*

28.2.2 Simplicity and Complexity

Complexity refers to the amount of resource required to perform a task. Begun in 1974 by Fagin, descriptive complexity has characterized all major notions of complexity in terms of the richness of logical languages needed to describe problems. [3]

[3] Given a description in some language and an object, the object is an interpretation of the description if there is a correspondence between the description and the object. The meaning of the text is provided by the correspondence between the description and the object. From Gödel's incompleteness theorem for arithmetic, we know that in general, objects cannot be fully described. It is trivial to demonstrate that each object may have multiple, mutually inconsistent descriptions. Consequently, we know, independently of any argument over hermeneutics, that the assertion that an interpretation is a authoritative requires objective justification.

Simplicity is the minimization of complexity. A parsimonious theory is the best known explanation, in the sense of Occam's Razor, of the infinite variety of alternative explanations for the same data. [4] Often it is the more modest or logically weaker theory. Typically, the logically weaker theory is a smaller theory i.e., fewer axioms. For theories there are two measures, number of axioms and length of proofs. Proof length is minimized when each theorem is an axiom. The number of axioms is minimized when

- each axiom is necessary (implies independent),
- the collection of axioms is sufficient to describe the properties of the domain and facilitate proofs, and
- the size of the theory (number of axioms) cannot be reduced through
  - generalization to combine axioms or
  - replacement by an alternate set of axioms.

Cohesion is the property that is used to distinguish one object or theory from another.

Definition 28.25 Cohesion. *Cohesion describes how well the contents of a theory cohere (stick together). It identifies those sentences that are logically connected to an application or a phenomena. A cohesive theory may be described through its interface and its details may be hidden (information hiding) from the larger context. The types of cohesion, in order of lowest to highest, are as follows:*

- *Coincidental cohesion - Coincidental cohesion is when parts of a module are grouped arbitrarily; the parts have no significant relationship (e.g. a module of frequently used functions).*
- *Logical cohesion - Logical cohesion is when parts of a module are grouped because of a slight relation (e.g. using control coupling to decide which part of a module to use, such as how to operate on a bank account).*
- *Temporal cohesion - Temporal cohesion is when parts of a module are grouped by when they are processed - the parts are processed at a particular time in program execution (e.g. a function which is called after catching an exception which closes open files, creates an error log, and notifies the user).*
- *Procedural cohesion - Procedural cohesion is when parts of a module are grouped because they always follow a certain sequence of execution (e.g. a function which checks file permissions and then opens the file).*

[4] For two or three millennia, logicians and mathematicians have been using the aesthetic values of abstractionism and minimalism to create their discipline. It is an esoteric form of fine art. Unlike the traditional work products of the fine art community, the work products of the mathematical community are produced cooperatively over time by many mathematicians.

- *Communicational cohesion - Communicational cohesion is when parts of a module are grouped because they operate on the same data (e.g. a method updateStudentRecord which operates on a student record, but the actions which method performs are not clear).*
- *Sequential cohesion - Sequential cohesion is when parts of a module are grouped because the output from one part is the input to another part (e.g. a function which reads data from a file and processes the data).*
- *Functional cohesion - Functional cohesion is when parts of a module are grouped because they all contribute to a single well-defined task of the module (a perfect module).*

http://en.wikipedia.org/wiki/Cohesion

Cohesion is maximized when the sentences in a theory are sufficient and each sentence is necessary.

Simple theories (ones composed of axioms) gain structure through lemmas, theorems, and corollaries. Further structure emerges when a collection of axioms, lemmas, theorems, and corollaries cohere to create a sub-theory.

Definition 28.26 Complex Theory. *A complex theory is a theory that is composed of sub-theories and a pattern of dependencies.*

Such a theory may be described as a many-sorted theory which consists of a collection of sub-theories with hierarchical or other organizational pattern.

Definition 28.27 Coupling. *Coupling describes how sub-theories interact. Highly coupled sub-theories have strong interconnections i.e. dependent on each other. Loosely coupled sub-theories are independent or almost independent. The types of coupling, in order of lowest to highest coupling, are as follows:*

- *Data coupling - Data coupling is when modules share data through, for example, parameters. Each datum is an elementary piece, and these are the only data which are shared (e.g. passing an integer to a function which computes a square root).*
- *Stamp coupling (Data-structured coupling) - Stamp coupling is when modules share a composite data structure, each module not knowing which part of the data structure will be used by the other (e.g. passing a student record to a function which calculates the student's GPA).*
- *Control coupling - Control coupling is one module controlling the logic of another, by passing it information on what to do (e.g. passing a what-to-do flag).*
- *Common coupling - Common coupling is when two modules share the same global data (e.g. a global variable).*

- *Content coupling - Content coupling is when one module modifies or relies on the internal workings of another module (e.g. accessing local data of another module).*

http://en.wikipedia.org/wiki/Coupling#Computer_programming

Loose coupling implies that there are no circular dependencies.

The complexity of a complex theory is minimized when

- each sub-theory is strongly cohesive,
- each sub-theory is necessary,
- the collection of sub-theories is sufficient,
- the collection of sub-theories is loosely coupled, and
- the size of the theory (number of sub-theories) cannot be reduced through
  - generalization to combine sub-theories or
  - replacement by an alternate set of sub-theories.

28.2.3 Categories of Theories

There are three basic types of theories; explanatory, predictive, and heuristic. The following is a taxonomy for empirical theories.

Let $\mathcal{T} = \langle A, I \rangle$ be a theory, F be a set of formulas, $F \in \overline{\mathcal{T}}$ $(\mathcal{T} \vdash F)$, and $\mathcal{M}_{\mathcal{T}}$ and $\mathcal{M}_F$ be structures where $\mathcal{M}_{\mathcal{T}} \models \mathcal{T}$ and $\mathcal{M}_F \models F$.

Predictive theory A theory $\mathcal{T}$ is a predictive theory for a set of formulas F of the form $A \to B$, if, for each formula $A \to B$ in F, $\mathcal{T} \vdash A \to B$, whenever $\mathcal{M} \models A$, then $\mathcal{M} \models B$ (the formulas have proofs and are also true).

The conditional form of the formulas is what sets predictive theories apart from explanatory theories and describes the conditions under which reliable results may be expected. A more descriptive term would be to call this type, contingent theories as the formulas may be read as "under the conditions described by A, B results". As may be expected, predictive theories are useful in planning and engineering as they produce predictable and reliable results.

Heuristic theory A theory $\mathcal{T}$ is a heuristic theory for a set of formulas F if $\mathcal{T} \vdash \mathcal{F}$ then it is decidable whether $\mathcal{M} \models \mathcal{F}$ (it is decidable whether theorems are true).

Heuristic theories are often speculative and controversial. The proofs may be sketchy at best. But because the conclusions are testable, the theory

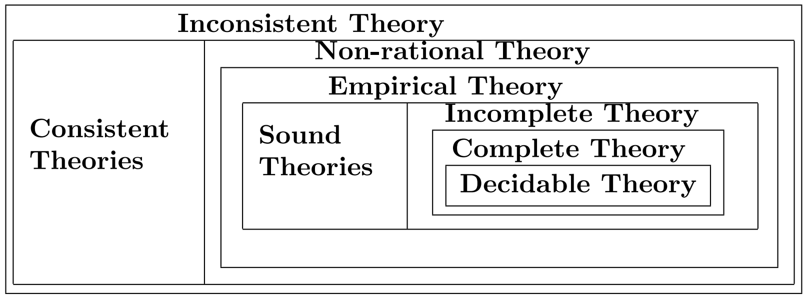

Figure 28.2: **A Theory Hierarchy**

contains a necessary element to suggest or stimulate research programs designed to confirm or deny the theory. Heuristic theories are useful in stimulating and encouraging further research, in particular, a search for the computable function which demonstrates the truth of the formula. If the formula is shown to be true, then it can become a part of a predictive theory. ... theories are based on a small descrete set of observations ... continuous function ... provides heuristic to suggest further observation. ... unsound inference rules ... ... application of metaphor ... ... application of a theory to a new domain ... ... flat (unstructured) memory space ...

Speculative theory A theory $\mathcal{T}$ is a speculative theory if the semantic relation is uncomputable or it is unknown to whether it is computable.

Explanatory theory A theory $\mathcal{T}$ is an explanatory theory for a set of formulas F if $M \models F$ and $T \vdash F$ (the formulas are true and have proofs). The proofs are the explanations. Note that the steps in the proof need not be sound. This is the situation with historical explanations such as theories of origins. Explanatory theories often provide a taxonomy that is useful for guiding observations, providing a language for descriptions, providing a base for new conceptions, and for drawing comparisons.

28.3 Epistemology and Theory Hierarchy

Epistemological methods are the methods that are used for the acquisition and justification of knowledge. Rationalism, empiricism, and non-rational methods are commonly used. The acquired knowledge permits the construction of relational structures and associated theories.

With respect to a taxonomy of theories, there are the three items of interest, the relational structure, the theory, and the relationship between the two. Figure 28.2 graphically displays the hierarchical relationship between three classes of consistent theories. An enclosed theory satisfies more restrictive conditions than the enclosing theory.

1. Each axiom must be independent of each other axiom.
2. Each axiom must be necessary.
3. The set of axioms (or axiom schema) must be finite.
4. Collectively the axioms must be sufficient to describe the system of interest. Consequently the axioms are consistent since they are satisfiable in at least one relational structure.
5. The inference rules must be sound (truth preserving) and effective.

Figure 28.3: **Quality Characteristics of Rational Theories**

28.3.1 Rationalism and Sound Theories

Rationalism is characterized by the use of sound rules of inference to create and justify knowledge. Mathematics and logic are prime examples of the use of the method. [5]

Hilbert's finite and effective conditions

Figure 28.3 provides a description of the distinguishing characteristics required by mathematical theories. These rules are open, effective, and objective making access to mathematics independent of language and culture. [6] Sound theories are precise, accurate, robust, reliable, objective, and universal. They are independent of gender, race, culture, and ideology.

Logics: propositional, predicate, modal

28.3.2 Empiricism and Empirical Theories

Empiricism is characterized by the use of observations by independent observers to justify and create knowledge. Figure 28.4 summarizes the quality characteristics of empirical theories.

> A theory is a good theory if it satisfies two requirements. It must accurately describe a large class of observations on the basis of a model that contains only a few arbitrary elements. And it must make definite predictions about the results of future observations.
> – Stephen Hawking [10]

[5] The fundamental concepts of mathematics are pure conceptual objects with no independent existence outside of the mind. Points, lines, planes, numbers, and functions are imaginary objects. And yet, without these imaginary objects human society would be reduced to a simple hunting gathering society. Purely fictitious objects provide the most mundane and utilitarian objects that make the life we enjoy possible.

[6] As a consequence of its emphasis on these values, the theories of mathematics are reliable, robust, objective (independent of time, place, person, and social group), universal, and independent of reality.

1. The formal theory must be a logical theory.
2. The semantic relation between the relational structure and the empirical domain must be observationally verifiable.

Figure 28.4: **Quality Characteristics of Empirical Theories**

$$\text{Theory} \quad \overset{\Delta}{\rightarrow} \quad \text{Abstract Domain} \quad \overset{O}{\rightarrow} \quad \text{Empirical Domain}$$

Where $\overset{O}{\rightarrow}$ designates an observational relationship between a well-defined abstract domain and an ill-defined empirical domain.

Figure 28.5: **Empirical Theory**

The natural sciences are the prime examples.

Scientific theories are explanations of physical phenomena that are required to have an effective relationship with natural phenomena. Since for each set of observations there is an infinite number of scientific theories, theory selection criteria include the aesthetic value of minimalism and heuristic value. The latter value often results in theories that lead to more questions than answers. The required effective relationship between theory and observation leads to theories that are statistically reliable, robust, objective, and universal.

The natural universe is complex, dynamic, and evolving. Its history spans generations of observers. Empirical theories are derived from and validated by observations. Empirical theories are the result of attempts to reverse engineer the empirical domain of interest. If a statement in an empirical theory is provable, it must be "observable". Validity is necessarily statistical in nature.

$$\text{Empirical: For all } P\text{, if } \vdash P \text{ then } \mathcal{M}_o \models P$$

where $\mathcal{M}_o \models P$ is determined by observation.
e.g. the Natural Sciences, software engineering, market research, etc.
Figure 28.5 illustrates the relationship between an empirical theory and the empirical domain of interest. The abstract domain is a relational structure that is an abstraction of the empirical domain. The abstract domain is constructed using empirical data and non-rational inference. Empirical theories are statistically improving in precision, accuracy, reliability, objectivity, and universality. They are independent of gender, race, culture, and ideology.

Logics: Belief revision

If the semantic relation is not computable (decidable), as it is in scientific theories, then:

> There never comes a point where a theory can be said to be true. The most that anyone can claim for any theory is that it has shared

> the successes of all its rivals and that it has passed at least one test which they have failed.
>
> – A. J. Ayer [2]

The complexity and inaccessibility of the empirical domain makes it is necessary to use unsound rules of inference to construct the relationship between the abstract domain and the empirical domain.

> Any physical theory is always provisional, in the sense that it is only a hypothesis; you can never prove it. No matter how many times the results of experiments agree with some theory, you can never be sure that the next time the result will not contradict the theory. On the other hand, you can disprove a theory by finding even a single observation that disagrees with the predictions of the theory.
>
> – Stephen Hawking [10]

Scientific theories are vague (within a margin of error) descriptions of the invariant (recurring) properties of a complex, dynamic universe. Due to the size and complexity of the universe and the margins of error, the theories cannot be known to be correct or complete. However, they are required to be objective and verifiable within the margin of error. Because of this, scientific theories are caricatures of reality rather than accurate and precise characterizations. Just as in a caricature, where the characteristic features of the subject represented are distorted, exaggerated, ennobled or beautified for effect, theories attempt to reveal the hidden ideal of the phenomena they seek to explain. *Speculative or forensic scientific theories* are the application of standard scientific theories to reconstruct non-recurring (historical) events or predict potential future events. Examples include cosmology, geology, evolutionary theory, forensic science.

28.3.3 Non-rational Epistemology and Theories

In many domains, the knowledge available is vague, incomplete, ambiguous, or inconsistent. Non-rational epistemology is characterized by the use of non-rational means to suggest and justify knowledge. [7] Often the theory is vague, incomplete, too specific, impossible (contradictory and not computable), and unverifiable. Figure 28.6 provides a description of the characteristics required to distinguish a non-rational theory from rational and empirical theories.

At a more fundamental level, the perceptual apparatus supplies patterns to the cognitive apparatus. The cognitive apparatus creates, maintains, and modifies the patterns and engages is pattern matching and pattern application activities [1]. The patterns are abstractions, generalizations, specializations, and

[7] Religion is a prime example of the method. Theories which develop in mystical traditions are dependent on subjective revelations (whether visions, meditation, supernatural revelation, or intuition) and have repeatedly demonstrated that they are unreliable, fragile, subjective (dependent on time, place, person, and social group), and tribal.

1. The set of axioms (or axiom schema) must be finite.
2. Collectively the axioms must be sufficient to describe the system of interest. Consequently the axioms are consistent since they are satisfiable in at least one relational structure.
3. The inference rules are not required to be sound.

Figure 28.6: **Quality Characteristics of Non-rational Theories**

$$\text{Theory} \overset{\Delta}{\rightarrow} \text{Abstract Domain} \overset{NR}{\rightarrow} \text{Inaccessible Domain}$$

Where $\overset{NR}{\rightarrow}$ designates a relationship between a well-defined abstract domain and an inaccessible domain.

Figure 28.7: **Non-rational Theory**

extensions.These fundamental activities and patterns provide support for non-rational epistemologies such as:

Mysticism: Information provided through a non-rational experience.
Superstition: Knowledge based on the assumption of a causal link between random events juxtaposed in time.
Random Choice: Knowledge (choice) determined by a random event such as a coin toss.
Imagination: A rational construction used to supply "missing information".
Intuition: Subconscious knowledge developed over time and derived from experience.

Non-rational theories (see Figure 28.7) are required when knowledge is vague, inaccurate, ambiguous, inconsistent, or incomplete. They are also required when there is no objective means of improving the knowledge. Validity is necessarily speculative or social in nature.

$$\text{Non-rational: For all } P\text{, if } \vdash P \text{ then } \mathcal{M}_{NR} \models P$$

where $\mathcal{M}_{NR} \models P$ is determined by a non-rational means such as mysticism, superstition, random choice, imagination, and intuition.
e.g., Custom, tradition, ideologies ...
Figure 28.7 illustrates the relationship between a non-rational theory and an inaccessible domain. Non-rational theories are often vague, inaccurate, fragile, unreliable, subjective, and tribal (ideological) since they are dependent on gender, race, culture, or ideology.
Logics: Fuzzy, Multi-valued, belief, possibility, plausibility

Myths are examples of non-rational theories. They are culturally significant stories or explanations which serve several purposes. The following listing of four types of myths is taken from [5] quoted in [4].

1. *Metaphysical myths* awaken and maintain an experience of awe, humility and respect in recognition of the ultimate mysteries of life and the universe.
2. *Cosmological myths* provide an image of the universe and explanations for how it works.
3. *Social myths* validate and help maintain an established social order.
4. *Psychological myths* support the centering and harmonization of the individual.

28.4 Axiology for Theories

There are a variety of lists of the characteristics of a good theory [10, 13, 15, 17]. Create ... apply ... modify ... communicate ... Axiology [9] (also called value theory) is the study of value or quality. Its focus is on how and what values determine actions. It includes ethics, aesthetics, political theory, and pragmatics. The extrinsic value of an object is the degree with which its properties correspond to a set of desired properties. A correct definition of goodness would be valuable because it might allow one to construct a good X by a reliable processes of deduction, elaboration or prioritization. It is important to distinguish between values and the mechanisms, techniques, and methods used to realize those values. Value can be computed. The value of an object is the degree to which it satisfies its defining conditions. Measurable values must be defined using the objects in the ontology. It is important to have a clear statement of values because then it might allow one to construct a theory that satisfies those values by some reliable process.

policy (value) vs mechanism (method); stakeholders - society

The desirable qualities of complex theories and software are similar. This observation leads to using the ISO Standard 9126 for Software Quality Characteristics as an organizing scheme for the values articulated for theories (See Figure 28.8). In the figure, compliance occurs in each value category. Compliance refers to compliance with established standards. For each quality characteristic X, the theory should comply with accepted standards for that characteristic even at the expense of an increase in complexity unless that complexity exceeds an acceptable threshold.

there are differences

Operational Qualities, Measures, and Methods

Functionality. (Subsection 28.2.3, Section 28.3) Users have different expectations for theories. The following are the most common.

Operational Qualities	**Maintenance Qualities**
• Functionality: suitability, accuracy, interoperability, security, compliance • Reliability: maturity, fault tolerance, recoverability compliance • Usability: understandability, learnability, operability, attractiveness, compliance • Efficiency: time behavior, resource utilization, compliance	• Maintainability: analyzability, changeability, stability, compliance • Portability: adaptability, installability, co-existence, replaceability, compliance

Figure 28.8: **ISO Standard 9126 Software Quality Characteristics**

- *Suitability* – Suitability is dependent on the domain of interest and the purpose of the theory. However, ...
  - *Multiple Connections* – Constructs must be multiply connected.
  - *Heuristic potential/New understanding* – The heuristic potential of a theory requires that the propositions that comprise the theory should suggest further hypotheses to be tested through research.

define generality and scope

  - *Scope/Generality* – The generality of a system is a measure of its usefulness and its credibility. Choose the theory that has a wider range of application i.e., choose $\mathcal{T}_2$ over $\mathcal{T}_1$ if P is a set of applications, $\mathcal{T}_2$ is more general than (logically stronger than) $\mathcal{T}_1$, and $P \cap \overline{\mathcal{T}_1} \subset P \cap \overline{\mathcal{T}_2}$. This may be accomplished semantically by choosing another semantic function or syntactically by modifying or adding axioms.
  - *Societal value* – The theory satisfies or creates curiosity. It helps to improve the human condition in some substantive way. It stimulates or generates change.
- *Accuracy (Precision + Validity)* – The validity of a theory depends on the relationship between the language and the relational structure and for empirical theories, the empirical domain. Up to the limitations of Gödel's incompleteness theorem, each proposition or its negation must be provable. Empirical theories require propositions to be testable or refutable through experiment and observations. Thus, some imaginable event, recognizable if it occurs, must suffice to refute the hypothesis.
- *Interoperability* – It is compatible with the giant joint hypothesis that is science as a whole. A new hypothesis may have to conflict with previous beliefs but the fewer the better.

Reliability. Theories with sound inference rules and axioms that are precise and valid are reliable. Simplicity is important for system reliability [18]. In particular, a simple reliable core component can ensure the critical properties of the system.

- *Maturity* – Reliability is an issue where the theory is derived using unsound inference rules which is the case of empirical theories. The theory is supported by many strands of evidence and survived many critical real world tests. The known reliability of an empirical theory depends on statistical information acquired through use.
- Independent of gender, race, culture, and ideology.
- *Fault tolerance*
  - Fault avoidance. Theories with sound inference rules are inherently reliable.
  - Fault tolerance through diversity.
  - Feedback control (forward recovery). Use the Simplex Architecture which consists of a simple high assurance subsystem and a complex high performance subsystem. The simple system is used whenever the complex system fails.

Usability. The usability of a theory depends on the cognitive abilities of the use, the expression of the theory, and the semantic relationship between the theory and the domain of interest. Simplicity is one of the most important properties for learnability [6, 7, 8]. It has been proposed as a unifying principle in cognitive science [6].

- *Attractiveness*/Aesthetic appeal – The theory should capture our interest and imagination. It should pique our curiosity about the phenomenon being explained.

Efficiency. The quality of efficiency has to do with minimizing the resource requirements for a task. Parsimony and simplicity facilitate efficiency in a variety of dimensions. An efficient description is one that requires few axioms. An efficient set of axioms is one that facilitates proofs. Equivalent theories may vary in the number of axioms and in the lengths of proofs.

Maintenance Qualities, Measures, and Methods

Theories change through *growth by intension*; changes to provide greater clarification of concepts. They also change through *growth by extension*; the addition of concepts.

Maintainability. Maintainability is facilitated if change can be confined to an easily identifiable component. Fewer components, Fewer interrelationships, Fewer implications. The following help to make a theory maintainable.

- *Analyzable* – A theory is analyzable if there is a decomposition of he theory into a set of independent components and the set of interactions between the components is well defined.
- *Changeability* – A theory is more changeable if the sub-theories are strongly coherent and the theory is loosely coupled with clearly defined dependencies. The effect of a change is determined by dependent relationships.
- *Stability* – The theory should be so constructed so that only significantly different and/or more accurate data is likely to cause it to be changed.

Portability. The portability of a theory with respect to X is enhanced if it satisfies the following characteristics.

- *Adaptability* – Abstract and generalized
- *Co-existence* – It is compatible with the larger collection of theories with which it must co-exist.
- *Replaceability* – A theory is replaceable if it has a clear well defined interface with the larger context in which it exists.
- it is expressed in a symbolic language with well-defined syntax and semantics and

28.5 Summary, Conclusions, and Suggestions for Further Research

This work makes several original contributions. The first is a descriptive rubric based on three areas of philosophy, ontology, epistemology, and axiology. It is employed to describe theories. The second is a hierarchical taxonomy for theories and a taxonomy for empirical theories. This taxonomy clarifies the relationships between theories in mathematics and science and theories in other disciplines. The third (which is developed in another paper) is the definition and use of basic metaphorical reasoning primitives as fundamental epistemological methods and as heuristics used in the evolution of theories. ... non-rational reasoning methods ... And last, is the use of the ISO Standard 9126 as a taxonomic framework for the values appropriate for theories.

... further research ... metaphorical reasoning ... necessary ... complete ... empirical evidence ... application to historical evolution in theories ...

Bibliography

[1] Aaby, Anthony A. *It is all about metaphor* 2004. `http://www.cs.wwc.edu/~aabyan/Articles/Metaphor.pdf`

[2] Ayer, A. J. *Philosophy in the Twentieth Century* 1982.

[3] Bostrom, Nick. *Anthropic Bias: Observation Selection Effects in Science and Philosophy.* Routledge 2002.

[4] Casti, John L. *Reality Rules: picturing the world in mathematics* vol. 2 *The frontier* Wiley Inter-Science 1992.

[5] Campbell, Joseph. *The Masks of God: Occidental Mythology* Viking, New York, 1964.

[6] Chater, N. and Vitanyi, P. "Simplicity: A unifying principle in cognitive science?" *Trends in Cognitive Sciences*, 7:1(2003), 19–22.

[7] Chater, Nick and Vitányi, Paul M. B. "The Generalized Universal Law of Generalization" *Journal of Mathematical Psychology*, 47:3(2003), 346–369. `homepages.cwi.nl/~paulv/papers/jmp02.pdf`

[8] Griffiths, Thomas L. and Tenenbaum, Joshua B. *Probability, algorithmic complexity, and subjective randomness* `web.mit.edu/cocosci/Papers/complex.pdf`

[9] Hartman, Robert S. *The Measurement of Value* `http://www.hartmaninstitute.org/html/MeasurementOfValue.htm`

[10] Hawking, Stephen. *A Brief History of Time* 1988.

[11] Immerman, Neil. *Descriptive Complexity* Springer Graduate Texts in Computer Science, 1999.

[12] Irvine, A. "Alfred Tarski", *Stanford Encyclopedia of Philosophy* Edward N. Zalta (ed.) at `http://plato.stanford.edu/entries/tarski-truth`

[13] Littlejohn, Stephen W. (1999) *Theories of Human Communication 6th ed.* Wadsworth Publishing Company 1999.

[14] Mahoney, Michael S. (2000) Software as Science - Science as Software in *Proceedings of the International Conference on History of Computing 2000 Mapping the History of Computing: Software Issues*Heinz Nisdorf Museums-Forum, Paderborn, Germany, 5-7 April 2000.

[15] Margenau, Henry (1950) *The nature of physical reality* McGraw-Hill Book Company, Inc.

[16] Peckhaus, Volker (1998) "The Heuristic Function of the Axiomatic Method." *20th World Congress of Philosophy.*

[17] Quine & Ullian (1988) Chapter 18 in *Introductory Readings in the Philosophy of Science*. Prometheus Books.

[18] Sha, Lui. *Using Simplicity to Control Complexity*

[19] "Theory" in *Wikipedia* `http://en.wikipedia.org/wiki/Theory`

Chapter 29

Harmonizing Conflicting Theories

Who is right?

Thoughts on harmonizing metaphysical systems

29.1 Introduction

Perhaps one of the most troubling aspects of metaphysics is that metaphysicians seem to be standing in a circle pointing to the metaphysician on the left and saying, ”I’m right and she is wrong” without any apparent means of resolving their differences. As Carnap put it, metaphysicians are not content just to present their systems, but they try to *refute* the metaphysical systems of others. This is the situation between evolutionists and creationists, encountered by software engineers in attempting to construct a consistent set of requirements for a software product, and in a committee attempting to resolve competing issues. Heliocentric vs geo-centric, light as waves vs light as particles & Heisenberg’s uncertainty principle is the result trying to use a particle model for a phenomena which does not quite fit the particle model. Perhaps a more enlightening approach would be to try to construct a new theory using what is true about each theory. Smullyan proposes the following tolerance principle: *Instead of trying to prove your opponent wrong, try to find out in what sense he may be right.* The resulting theory would be more complete then either theory alone and permit access to new theorems not accessible from either theory.

29.2 Conflicting world views

Note: This section is an adaptation of material from R. M. Smullyan in *5000 B.C. and other Philosophical Fantasies*. We begin with a highly idealized approach to the differences between metaphysicians and then move to a more realistic position that suggests a methodology for resolving their differences. Without loss of generality, we formalize the situation for two metaphysicians with two distinct and incompatible world views. But first, we define a perfect world view.

Perfect world view A *perfect world view* for a person, p, is a world view, W, which is internally consistent, consistent with experience, consistent with feelings and intuition, and (most certainly unrealistic) consistent with all future experience for person p.

Metaphysical world view A
Let a be a person with perfect world view A (logically incompatible with B).

Metaphysical world view B
Let b be a person with perfect world view B (logically incompatible with A).

The two world views may be assumed to share the same words but not necessarily with the same meaning. Arguments between a and b over their respective world views indicates that each believes that the other's world view is not perfect and hopes to either show that the other view is inconsistent or produce some new experience in the other that will change his/her mind. However, we are assuming that this is not possible and so the essence of their differences is captured by the addition of a single statement to their respective world views.

Metaphysical world view A'
A' is A plus the statement "World view A is *true* of the real world and world view B is *false* of the real world."

Metaphysical world view B'
B' is B plus the statement "World view B is *true* of the real world and world view A is *false* of the real world."

Now, if both world views A' and B' are consistent, there is no way for a or b to show the other to be wrong. The refutation of a system requires some verifiable correspondence with the real world even then, it is not the world view that can be disproved or discredited, it is only the correspondence relation between the world view and the natural world that may be discredited. However, it is possible for the two metaphysicians a and b to learn from each other. Smullyan suggests a tolerance principle.

Tolerance Principle (Smullyan) Instead of trying to prove your opponent wrong, try to find out in what sense (s)he may be right.

Making an analogy to mathematical logic, there is *some* interpretation of all the terms in a perfect world view that is true. According to this there is a

reinterpretation of the language such that every thing that is said to be true in a perfect world view is true about the real world. This approach requires the extension of one world view with an interpretation of another world view.

Metaphysical world view A''	**Metaphysical world view** B''
A'' is A plus an interpretation of world view B is *true* of the real world."	B'' is B plus an interpreation of world view A is *true* of the real world."

The extended world view allows each to be able to see the world through the eyes of the other, permits each to extend their world views, and possibly decide hitherto undecidable propositions.

Suppose a world view is not perfect – it is inconsistent. In an inconsistent world view, all statements are both true and false (there are ways around this rule). So, it would appear that such a world view is worthless. In reality, no world view is presented as an unstructured collection of statements. In an analogy with mathematical logic, some statements are axioms (deliberately chosen to be independent of each other) and the remaining statements are derived from the axioms by truth preserving rules of inference. So the challenge is to massage a collection of statements into a consistent world view by identifying and deleting or correcting faulty axioms or identifying faulty arguments and either dropping conclusions derived using faulty logic or supplying valid arguments. Smullyan's tolerance principle is of use even with inconsistent world views, incomplete world views, or world views whose consistency is unknown.

A formal approach

A more formal approach is the following. Let Σ be a possibly countably infinite set of symbols and $L(\Sigma)$ be a language (a set of sentences or strings), defined using the symbols in Σ, for $M = \langle S, I\rangle$ a *structure* for the language where S is a non-empty set and I is correspondence between the sentences of the language and S. For a sentence, Φ, of the language, M is a *model* of Φ, written $M \models \Phi$, if the sentence corresponds to an object in the model M. i.e., $I(\Phi) \in S$. A *theory* is a set of sentences and for a given theory T, $M \models T$ denotes that $M \models \Phi$ for each sentence Φ in T.

Aside: In the arithmetic of the natural numbers, the structure simple and is easily axiomatized, S is just the number sequence with the operations of addition and multiplication. Many interesting properties are emergent. For example, prime numbers and perfect numbers are emergent properties of the number sequence and the operations of addition and multiplication. In science, the structure is complex and largely unknown and the correspondence relation is subjected to experimental verification. In metaphysics, the structure is speculative and inaccessible and the correspondence relation is unverifiable.

Let T_A and T_B be two distinct theories in $L(\Sigma)$ with structures $M_A = \langle S, I_A\rangle$ and $M_B = \langle S, I_B\rangle$ respectively. Assume that the theories, while individually

consistent and closed under some rules of inference, are incompatible.

- A theory is said to be *consistent* if it is a proper subset of $L(\Sigma)$, i.e., does not contain all the formulas in $L(\Sigma)$.
- A theory is said to be *closed under some rules of inference* if it contains all formulas inferred by the application of the rules of inference to the formulas in the theory. A formula f inferred from the formulas in a theory T is written $T \vdash f$. *Sound* rules of inference are truth preserving so that if a formula f has a proof, it is also true in M, that is, if $T \vdash f$, then $M \models f$.
- T_A and T_B are said to be *incompatible* if there is a formula f in one theory and its negation, $\neg f$, in the other such that, $M_A \models f$ and $M_B \models \neg f$.

Since the theories are incompatible, both $I_A(f)$ and $I_B(\neg f)$ cannot be true and therefore must be undefined, uncomputable, or unverifiable.

In every theory, whether it is mathematical, scientific, or metaphysical, we may assume that there are three classes of statements

1. Statements that have proofs in the theory and have been verified to hold in the model.
2. Statements that have proofs in the theory but have not been verified to hold in the model.
3. Statements that do not have proofs.

Statements of the first type are true but may have a better interpretation. Statements of the second and third types may be contested by demonstrating a counter example. However, in the case of metaphysical theories, the statements may be neither verifiable nor have counter examples.

One approach to utilizing Smullyan's tolerance principle is to use a transformation function $\tau \in L(\Sigma) \rightarrow L(\Sigma)$ (extended to apply to sets) which transforms one formula to another by say, marking each formula with the identity of the theory to which it belongs. Contradictory statements in different theories are no longer contradictory because they are tagged with the theory from which they are derived. This approach resolves the contradiction so that $\tau(T_B)$ is compatible with T_A. This approach is commonly used by religious fundamentalists to resolve apparent contradictions which arise from literalist interpretations or conflicts with scientific data.

The following is an open question.

Open Question Given two incompatible theories, determine the conditions under which transformation functions exist that preserve the proof and truth relations in each theory and remove the incompatibilities.

A more modest approach is sketched out in the next section which gives each theory the right to make positive statements but denies each theory the right to make negative statements concerning the other theory.

29.3 A limited approach to merging theories

Consistent theories are neither true nor false. A theory is true or false with respect to an interpretation in a structure. So the question is, given a theory and a structure, is there an interpretation between them that makes the theory true in the structure?

There are a range of methods for combining theories.

1. Merge theories that do not conflict. The theories may or may not have the same language.
2. Rename propositions or drop propositions when theories conflict (i.e. for some proposition p, both p and ¬p are present):
   - Rename the conflicting propositions throughout the theories. This option maintains the size of both theories. A proposition in one theory may be mapped to a compatible proposition in the other theory (syncretism) or to a proposition not present in either theory. This approach retains both theories and increases the size and completeness of a theory.
   - Drop one or both of the conflicting propositions (reduces the size and completeness of a theory). See Table 1.
     (a) Drop the conflicting negative propositions. This is a pluralistic position. It is typical of creative, brainstorming sessions and feature creep in software. In Table 1, these are in the blue cell.
     (b) Drop the conflicting positive propositions. This is a conservative position which gives each theory veto power over the other. It is typical of limiting, stifling environments. In Table 1, these are in the yellow cell.
     (c) Drop the conflicting propositions (both). This position permits only the common core. In software engineering, it is the position that identifies those features to be included in the initial release. In Table 1, these are in the red cells.

From the perspective of logic, these are reasonable methods as the propositions do not have any meaning but just truth values.

Harmonizing Propositional Theories

Given two consistent theories in propositional logic, is there a way to combine the theories so that the resulting theory is consistent? The simplest way is to

	Drop both		Drop positives		Drop negatives	
Positive propositions	R		R		R	
Conflicting propositions	P	¬P	P	¬P	P	¬P
Negative propositions	¬S		¬S		¬S	

Figure 29.1: Dropped propositions in merging theories

simply map the theories to different languages and take the union of the theories. If the theories share a common language, one theory could be mapped to a different language or to different propositions in the same language. A syncretic approach would be to map the propositions of one theory to propositions of the other.

This work assumes that both theories are in the same language and a harmonizing valuation function is constructed from the valuation functions of the given theories. The basic idea is that the harmonized theory contains those positive propositions that at least one theory affirms and denies those propositions that both theories deny or, alternately, affirm those propositions that both affirm and deny those that at least one denies. The first approach is similar to trying to use all good ideas and is the source of *feature creep* in software. The second approach is to use only those ideas on which there is full agreement.

Since propositional logic is decidable, each theory may be represented by a valuation function (a boolean valued function on the propositions and logical formulas). The following shows how to construct a valuation function (called the harmonizing function) from two other valuation functions. In what follows, we assume that logical formulas are in either the conjunctive or disjunctive normal form.

The Details

Definition 1 - Syntax Let P be a non-empty set and L a set defined as follows:

1. $P \subset L$
2. If $p \in P$, then $p \notin L$
3. If A, $B \in L$, then $A \wedge B$ and $A \vee B \in L$
4. Rules 1-3 define L.

Definition 2 - Semantics Let $V \subset L \rightarrow \{0,1\}$ where for every $v \in V$, $p \in P$, and $\neg p$, $A \wedge B$, $A \vee B \in L$

1. $v(p) \in \{0,1\}$

$v_1(p)$	$v_2(p)$	$v_1(\neg p)$	$v_2(\neg p)$	$\min(v_1(\neg p), v_2(\neg p))$	$\max(v_1(p), v_2(p))$	$1 - \max(v_1(p), v_2(p))$
1	1	0	0	0	1	0
1	0	0	1	0	1	0
0	0	1	0	0	1	0
0	0	1	1	1	0	1

Since the two relevant columns are the same, we have equality. QED.

Figure 29.2: Truth table for $\min(v_1(\neg p), v_2(\neg p)) = 1 - \max(v_1(p), v_2(p))$

2. $v(\neg p) = 1 - v(p)$
3. $v(A \wedge B) = \min(v(A), v(B))$
4. $v(A \vee B) = \max(v(A), v(B))$

The set V is the set of *Boolean valuation functions* of propositional logic.

Proposition 1 There is at least one valuation function i.e., $V \neq \emptyset$. Proof: left to the reader

Observation: Valuation functions may be created from sets of formulas designated to be true formulas. These sets of formulas are called theories. These sets must obey the following constraints:

1. Exactly one of the pair $(p, \neg p)$ is in the set.
2. If $A \wedge B$ is in the set, so are A and B in the set.
3. If $A \vee B$ is in the set, then at least one of A and B is in the set.

These constraints ensure that the theory is consistent.

Definition 3 - Harmonizing functon Let $v_1, v_2 \in V$, $p \in P$, $\neg p$, A, B $\in L$, and define $v_h \in L \rightarrow \{0, 1\}$ as follows

1. $v_h(p) = \max(\ v_1(p), v_2(p)\ )$
2. $v_h(\neg p) = \min(\ v_1(\neg p), v_2(\neg p)\ )$
3. $v_h(A \wedge B) = \min(v_h(A), v_h(B))$
4. $v_h(A \vee B) = \max(v_h(A), v_h(B)$

Proposition 2: The harmonizing functions are Boolean valuation functions i.e., $v_h \in V$.

Proof: Conditions 1, 3, and 4 of definition 3 satisfy conditions 1, 3, and 4 of definition 2. We must show that condition 2 of definition 3 satisfies condition 2 of definition 2. That is, we must show, $v_h(\neg p) = 1 - v_h(p)$ or equivalently by the truth table below

Theorem: Suppose we have two theories (i.e., two subsets of L - sets of formulas in propositional logic). If there are valuation functions that make each formula in each set true, the harmonizing function constructed from the two valuation functions makes the *harmonizing union* of the theories true.

Proof: left to the reader.

Examples and Comments

Here are three example to illustrate the use of the harmonizing function.

Suppose the following two positive propositions are in different theories:

- A: The earth was created 6,000 years ago.
- B: The earth was created 1,000,000,000 years ago.

A harmonizing valuation function would accept both statements. Remember that in propositional logic, all we know is the truth value of a proposition thus it is not possible to recognize the inconsistency that occurs when both propositions are true.

Suppose two theories share a negative proposition like the following:

- It is false that the earth is flat.

In this case, the harmonizing function would accept the proposition.

Suppose two theories hold contradictory positions, for example, the positive proposition A is in one theory and its negation, $\neg$A is in the other theory like the following:

- A: God exists.
- $\neg$ A: God does not exist.

The harmonizing function would accept the positive proposition and reject the negative proposition.

These examples illustrate the action of the harmonizing function and demonstrate that, as in the first example, that propositional logic is not adequate to resolve many interesting questions. Propositional logic is only satisfactory for use when the propositions are not in conflict on some other level of meaning. Clearly the trick, to get propositions accepted, is to make positive statements.

Harmonizing first-order theories

Given two consistent theories in first-order logic, they may be combined to produce a third consistent theory by simply mapping the theories to different

languages and take the union of the theories. In this work, a harmonizing valuation function is constructed from the valuation functions of the given theories. The basic idea is that the harmonized theory contains those positive propositions that at least one theory affirms and denies those propositions that both theories deny.

Since propositional logic is decidable, each theory may be represented by a valuation function (a boolean valued function on the propositions and logical formulas). The following shows how to construct a valuation function (called the harmonizing function) from two other valuation functions. In what follows, we assume that logical formulas are in either the conjunctive or disjunctive normal form.

The Details

Definition 1 - Syntax Let T be a non-empty set of terms, P be a non-empty set and L a set defined as follows:

1. P⊂ L
2. If p(a)∈P, then ¬p(a)∈L
3. If A, B ∈ L, then A∧B and A∨B ∈ L
4. If Q(x)∈L, then ∀x Q(t)[x/t]∈L and ∃x Q(t)[x/t]∈L where t∈T.
5. Rules 1-3 define L.

Definition 2 - Semantics Let V ⊂ L→0,1 where for every v∈V, p∈P, and ¬p, A∧B, A∨B, ∀x Q(x), ∃x Q(x) ∈L

1. v(p) ∈0,1
2. v(¬p) = 1 - v(p)
3. v(A∧B) = min(v(A), v(B))
4. v(A∨B) = max (v(A), v(B))
5. v(∀x Q(x)) = ¡math $\min_{t\in T}$v(Q(x)[t/x])
6. v(∀x Q(x)) = $\max_{t\in T}$v(Q(x)[t/x])

The set V is the set of *Boolean valuation functions* of first-order logic.

Proposition 1 There is at least one valuation function i.e., V∈ ∅

Proof: left to the reader

Observation: Valuation functions may be created from sets of formulas designated to be true formulas. These sets of formulas are called theories. These sets must obey the following constraints:

1. Exactly one of the pair (p, ¬p) is in the set.
2. If A∧B is in the set, so are A and B in the set.

v_1(p)	v_2(p)	$v_1(\neg$p)	$v_2(\neg$p)	min($v_1(\neg$p), $v_2(\neg$p))	max(v_1(p), v_2(p))	1 - max(v_1(p), v_2(p))
1	1	0	0	0	1	0
1	0	0	1	0	1	0
0	0	1	0	0	1	0
0	0	1	1	1	0	1

Since the two relevant columns are the same, we have equality. QED.

Figure 29.3: Truth table for min($v_1(\neg$p), $v_2(\neg$p)) = 1 - max(v_1(p), v_2(p))

3. If A∧B is in the set, then at least one of A and B is in the set.
4. If ∀x Q(x) is in the set, then so is Q(x)[a/x] for all t∈T.
5. If ∃x Q(x) is in the set, then so is Q(x)[a/x] for some t∈T.

These constraints ensure that the theory is consistent.

Definition 3 - Harmonizing function Let v_1, v_2 ∈V, p∈P, ¬p, A, B ∈L, and define v_h∈L→0, 1 as follows

1. v_h(p) = max(v_1(p), v_2(p))
2. $v_h(\neg$p) = min($v_1(\neg$p), $v_2(\neg$p))
3. v_h(A∧B) = min(v_h(A), v_h(B))
4. v_h(A∨B) = max(v_h(A), v_h(B)
5. v_h(∀x Q(x)) = $\min_{t \in T}$$v_h$(Q(x)[t/x])
6. v_h(∀x Q(x)) = $\max_{t \in T}$$v_h$(Q(x)[t/x])

Proposition 2: The harmonizing functions are Boolean valuation functions i.e., $v_h \in V$.
Proof: Conditions 1, 3, and 4 of definition 3 satisfy conditions 1, 3, and 4 of definition 2. We must show that condition 2 of definition 3 satisfies condition 2 of definition 2. That is, we must show, $v_h(\neg p) = 1 - v_h(p)$ or equivalently by the truth table below:

Theorem: Suppose we have two theories (i.e., two subsets of L - sets of formulas in first-order logic). If there are valuation functions that make each formula in each set true, the harmonizing function constructed from the two valuation functions makes the *harmonizing union* of the theories true.
Proof: left to the reader.

Examples and Comments

29.4 Summary and Conclusions

29.5 References

For an abstract illustration of harmonizing incompatible theories see: Smullyan, Raymond M. in *5000 B.C. and other Philosophical Fantasies*

For details of propositional logic see: Smullyan, Raymond M. Chapter 1 in *First-Order Logic*

For details of first-order logic see: Smullyan, Raymond M. Chapters 4 & 5 in *First-Order Logic*

Chapter 30

Vagueness

by
Bertrand Russell
First published in
The Australasian Journal of Psychology and Philosophy, 1 (June 1923): 84–92
This text is from *Collected Papers,* vol. 9, pp. 147–154.
Note: 'Vague' is now called 'fuzzy' and emphasis mine - A. Aaby

Reflection on philosophical problems has convinced me that a much larger number than I used to think, or than is generally thought, are connected with the principles of symbolism, that is to say, with the relation between what means and what is meant. In dealing with highly abstract matters it is much easier to grasp the symbols (usually words) than it is to grasp what they stand for. The result of this is that almost all thinking that purports to be philosophical or logical consists in attributing to the world the properties of language. Since language really occurs, it obviously has all the properties common to all occurrences, and to that extent the metaphysic based upon linguistic considerations may not be erroneous. But language has many properties which are not shared by things in general, and when these properties intrude into our metaphysic it becomes altogether misleading. I do not think that the study of the principles of symbolism will yield a any *positive* results in metaphysics, but I do think it will yield a great many negative results by enabling us to avoid fallacious inferences from symbols to things. The influence of symbolism on philosophy is mainly unconscious; if it were conscious it would do less harm. By studying the principles of symbolism we can learn not to be unconsciously influenced by language, and in this way can escape a host of erroneous notions.

Vagueness, which is my topic tonight, illustrates these remarks. You will no doubt think that, in the words of the poet: "Who speaks of vagueness should himself be vague." I propose to prove that all language is vague, and that therefore my language is vague, but I do not wish this conclusion to be one

that you could derive without the help of the symbolism. I shall be as little vague as I know how to be if I am to employ the English language. You all know that I invented a special language with a view to avoiding vagueness, but unfortunately it is unsuited for public occasions. I shall, therefore, though regretfully, address you in English, and whatever vagueness is to be found in my words must be attributed to our ancestors for not having been predominantly interested in logic.

There is a certain tendency in those who have realized that words are vague to infer that things also are vague. We hear a great deal about the flux and the continuum and the unanalysability of the Universe, and it is often suggested that as our language becomes more precise, it becomes less adapted to represent the primitive chaos out of which man is supposed to have evolved the cosmos. This seems to me precisely a case of the fallacy of verbalism–the fallacy that consists in mistaking the properties of words for the properties of things. Vagueness and precision alike are characteristics which can only belong to a representation, of which language is an example. They have to do with the relation between a representation and that which it represents. Apart from representation, whether cognitive or mechanical, there can be no such thing as vagueness or precision; things are what they are, and there is an end of it. Nothing is more or less what it is, or to a certain extent possessed of the properties which it possesses. Idealism has produced habits of confusion even in the minds of those who think that they have rejected it. Ever since Kant there has been a tendency in philosophy to confuse knowledge with what is known. It is thought that there must be some kind of identity between the knower and the known, and hence the knower infers that the known is also muddle-headed. All this identity of knower and known, and all this supposed intimacy of the relation of knowing, seems to me a delusion. Knowing is an occurrence having a certain relation to some other occurrence, or groups of occurrences, or characteristic of a group of occurrences, which constitutes what is said to be known. When knowledge is vague, this does not apply to the knowing as an occurrence; as an occurrence it is incapable of being either vague or precise, just as all other occurrences are. Vagueness in a cognitive occurrence is a characteristic of its relation to that which is known, not a characteristic of the occurrence in itself.

Let us consider the various ways in which common words are vague, and let us being with such a word as "red". It is perfectly obvious, since colours form a continuum, that there are shades of colour concerning which we shall be in doubt whether to call them red or not, not because we are ignorant of the meaning of the word "red", but because it is a word the extent of whose application is essentially doubtful. This, of course, is the answer to the old puzzle about the man who went bald. It is supposed that at first he was not bald, that he lost his hairs one by one, and that in the end he was bald; therefore, it is argued, there must have been one hair the loss of which converted him into a bald man. This, of course, is absurd. Baldness is a vague conception; some men are certainly bald, some are certainly not bald, while

between them there are men of whom it is not true to say they must be either be bald or not bald. *The law of excluded middle is true when precise symbols are employed, but it is not true when symbols are vague, as, in fact, all symbols are.*

All words denoting sensible qualities have the same kind of vagueness which belongs to the word “red”. This vagueness exists also, though in a lesser degree, in the quantitative words which science has tried hardest to make precise, such as a metre or a second. I am not going to invoke Einstein for the purpose of making these words vague. The metre, for example, is defined as the distance between two marks on a certain rod in Paris, when that rod is at a certain temperature. Now the marks are not points, but patches of a finite size, so that the distance between them is not a precise conception. Moreover, temperature cannot be measured with more than a certain degree of accuracy, and the temperature of a rod is never quite uniform. For all these reasons the conception of a metre is lacking in precision. The same applies to a second. The second is defined by relation to the rotation of the earth, but the earth is not a rigid body, and two parts of the earth's surface do not take exactly the same time to rotate; moreover all observations have a margin of error. There are some occurrences of which we can say that they take less than a second to happen, and others of which we can say that they take more, but between the two there will be a number of occurrences of which we believe that they do not all last equally long, but of none of which we can say whether they last more or less than a second. Therefore, when we say an occurrence lasts a second, all that it is worth while to mean is that no possible accuracy of observation will show whether it lasts more or less than a second.

Now let us take proper names. I pass by the irrelevant fact that the same proper name often belongs to many people. I once knew a man called Ebenezer Wilkes Smith, and I decline to believe that anybody else ever had this name. You might say, therefore, that here at last we have discovered an unambiguous symbol. This, however, would be a mistake. Mr. Ebenezer Wilkes Smith was born, and being born is a gradual process. It would seem natural to suppose that the name was not attributable before birth; if so, there was doubt, while birth was taking place, whether the name was attributable or not. If it be said that the name was attributable before birth, the ambiguity is even more obvious, since no one can decide how long before before the name became attributable. Death is also a process: even when it is what is called instantaneous, death must occupy a finite time. If you continue to apply the name to the corpse, there must gradually come a stage in decomposition when the name ceases to be attributable, but no one can say precisely when this stage has been reached.

The fact is that all words are attributable without doubt over a certain area, but become questionable within a penumbra, outside which they are again certainly not attributable. Someone might seek to obtain precision in the use of words by saying that no word is to be applied in the penumbra, but unfortunately the penumbra is itself not accurately definable, and all the

vaguenesses which apply to the primary use of words apply also when we try to fix a limit to their indubitable applicability. This has a reason in our physiological constitution. Stimuli which for various reasons we believe to be different produce in us indistinguishable sensations. It is not clear whether the sensations themselves are sometimes identical in relevant respects even when the stimuli differ in relevant respects. This is a kind of question which the theory of quanta at some much later stage in its development may be able to answer, but for the present it may be left in doubt. For our purpose it is not the vital question.

What is clear is that the knowledge that we can obtain through our sensations is not as fine-grained as the stimuli to those sensations. We cannot see with the naked eye the difference between two glasses of water of which one is wholesome while the other is full of typhoid bacilli. In this case a microscope enables us to see the difference, but in the absence of a microscope the difference is only inferred from the differing effects of things which are sensibly indistinguishable. It is this fact that things which our senses do not distinguish produce different effects–as, for example, one glass of water gives you typhoid while the other does not–that has led us to regard the knowledge derived from the senses as vague. And the vagueness of the knowledge derived from the senses infects all words in the definition of which there is a sensible element. This includes all words which contain geographical or chronological constituents, such as “Julius Caesar”, “the twentieth century”, or “the solar system”.

There remains a more abstract class of words: first, words which apply to all parts of time and space, such as “matter” or “causality”; secondly, the words of pure logic. I shall leave out of discussion the first class of words, since all of them raise great difficulties, and I can scarcely imagine a human being who would deny that they are all more or less vague. I come therefore to the words of pure logic, words such as “or” and “not”. Are these words also vague or have they a precise meaning?

Words such as “or” and “not” might seem, at first sight, to have a perfectly precise meaning: “p or q” is true when p is true, when q is true, and false when both are false. But the trouble is that this involves the notions of “true” and “false”; and it will be found, I think, that all the concepts of logic involve these notions, directly or indirectly. Now “true” and “false” can only have a *precise* meaning when the symbols employed–words, perceptions, images, or what not–are themselves precise. We have seen that, in practice, this is not the case.

It follows that *every proposition that can be framed in practice has a certain degree of vagueness; that is to say, there is not one definite fact necessary and sufficient for its truth, but a certain region of possible facts, any one of which would make it true. And this region is itself ill-defined*: we cannot assign to it a definite boundary. This is the difference between vagueness and generality. A proposition involving a general concept–e.g. “This is a man”–will be verified by a number of facts, such as “This” being Brown or Jones or Robinson. But

if "man" were a precise idea, the set of possible facts that would verify "this is a man" would be quite definite. Since, however, the conception "man" is more or less vague, it is possible to discover prehistoric specimens concerning which there is no, even in theory, a definite answer to the question "Is this a man?" As applied to such specimens, the proposition "this is a man" is neither definitely true nor definitely false. Since all non-logical words have this kind of vagueness, it follows that the conceptions of truth and falsehood, as applied to propositions composed of or containing non-logical words, are themselves more or less vague.

Since propositions containing non-logical words are the substructure on which logical propositions are built, it follows that logical propositions also, so far as we can know them, become vague through the vagueness of "truth" and "falsehood". We can see an ideal of precision, to which we can approximate indefinitely; but we cannot attain this ideal. Logical words, like the rest, when used by human beings, share the vagueness of all other words. There is, however, less vagueness about logical words than about the words of daily life, because logical words apply essentially to symbols, and may be conceived as applying rather to possible than to actual symbols. We are capable of imagining what a precise symbolism would be, though we cannot actually construct such a symbolism. Hence we are able to *imagine* a precise meaning for such words as "or" and "not". We can, in fact, see precisely what they would mean if our symbolism were precise.

All traditional logic habitually assumes that precise symbols are being employed. It is therefore not applicable to this terrestrial life, but only to an imagined celestial existence. Where, however, this celestial existence would differ from ours, so far as logic is concerned, would be not in the nature of what is known, but only in the accuracy of our knowledge. Therefore, if the hypothesis of a precise symbolism enables us to draw any inferences as to what is symbolized, there is no reason to distrust such inferences merely on the ground that our actual symbolism is not precise. We are able to conceive precision; indeed, if we could not do so, we could not conceive vagueness, which is merely the contrary of precision. This is one reason why logic takes us nearer to heaven than most other studies. On this point I agree with Plato. But those who dislike logic will, I fear, find my heaven disappointing.

It is now time to tackle the definition of vagueness. Vagueness, though it applies primarily to what is cognitive, is a conception applicable to every kind of representation–for example, a photograph, or a barograph. But before defining vagueness it is necessary to define accuracy. One of the most easily intelligible definitions of accuracy is as follows: One structure is an accurate representation of another when the words describing the one will also describe the other by being given new meanings. For example, "Brutus killed Caesar" has the same structure as "Plato loved Socrates", because both can be represented by the symbol "xRy", by giving suitable meanings to x and R and y. But this definition, though easy to understand, does not give the essence of the matter, since the introduction of words describing the two systems is

irrelevant. The exact definition is as follows: *One system of terms related in various ways is an accurate representation of another system of terms related in various other ways if there is a one-one relation of the terms of the one to the terms of the other, and likewise a one-one relation of the relations of the one to the relations of the other, such that, when two or more terms in the one system have a relation belonging to that system, the corresponding terms of the other system have the corresponding relation belonging to the other system.* Maps, charts, photographs, catalogues, etc. all come within this definition in so far as they are accurate.

Per contra, a representation is vague when the relation of the representing system to the represented system is not one-one, but one-many. For example, a photograph which is so smudged that it might equally represent Brown or Jones or Robinson is vague. A small-scale map is usually vaguer than a large-scale map, because it does not show all the turns and twists of the roads, rivers, etc. so that various slightly different courses are compatible with the representation that it gives. Vagueness, clearly, is a matter of degree, depending upon the extent of the possible differences between different systems represented by the same representation. Accuracy, on the contrary, is an ideal limit.

Passing from representation in general to the kinds of representation that are specially interesting to the logician, the representing system will consist of words, perceptions, thoughts, or something of the kind, and *the would-be one-one relation between the representing system and the represented system will be* meaning. In an accurate language, meaning would be a one-one relation; no word would have two meanings, and no two words would have the same meaning. In actual languages, as we have seen, meaning is one-many. (It happens often that two words have the same meaning, but this is easily avoided, and can be assumed not to happen without injuring the argument.) That is to say, there is not only one object that a word means, and not only one possible fact that will verify a proposition.

The fact that meaning is a one-many relation is the precise statement of the fact that all language is more or less vague. There is, however, a complication about language as a method of representing a system, namely that words which mean relations are not themselves relations, but just as substantial or unsubstantial as other words. In this respect a map, for instance, is superior to language, since the fact that one place is to the west of another is represented by the fact that the corresponding place on the map is to the left of the other; that is to say, a relation is represented by a relation. But in language this is not the case. Certain relations of higher order are represented by relations, in accordance with the rules of syntax. For example, "A precedes B" and "B precedes A" have different meanings, because the order of the words is an essential part of the meaning of the sentence. But this does not hold of elementary relations; the word "precedes", though it means a relation, is not a relation. I believe that this simple fact is at the bottom of the hopeless muddle which has prevailed in *all* schools of philosophy as to the

nature of relations. It would, however, take me too far from my present theme to pursue this line of thought.

It may be said: How do you know that all knowledge is vague, and what does it matter if it is? The case which I took before, of two glasses of water, one of which is wholesome while the other gives you typhoid, will illustrate both points. Without calling in the microscope , it is obvious that you cannot distinguish the wholesome glass of water from the one that will give you typhoid, just as, without calling in the telescope, it is obvious that what you see of a man who is 200 yards away is vague compared to what you see of a man who is 2 feet away; that is to say, many men who look quite different when see close at hand look indistinguishable at a distance, while men who look different at a distance never look indistinguishable when seen close at hand. Therefore, according to the definition, there is less vagueness in the near appearance than in the distant one. There is still less vagueness about the appearance under the microscope. It is perfectly ordinary facts of this kind that prove the vagueness of most of our knowledge, and lead us to infer the vagueness of all of it.

It would be a great mistake to suppose that vague knowledge must be false. On the contrary, *a vague belief has a much better chance of being true than a precise one, because there are more possible facts that would verify it.* If I believe that so-and-so is tall, I am more likely to be right than if I believe that his height is between 6 ft. 2 in. and 6 ft. 3 in. In regard to beliefs and propositions, though not in regard to single words, we can distinguish between accuracy and precision. *A belief is* precise *when only one fact would verify it; it is* accurate *when it is both precise and true.* Precision diminishes the likelihood of truth, but often increases the pragmatic value of a belief if it is true–for example, in the case of the water that contained the typhoid bacilli. *Science is perpetually trying to substitute more precise beliefs for vague ones; this makes it harder for a scientific proposition to be true than for the vague beliefs of uneducated persons to be true, but it makes scientific truth better worth having if it can be obtained.*

Vagueness in our knowledge is, I believe, merely a particular case of a general law of physics, namely that law that what may be called the appearances of a thing at different places are less and less differentiated as we get further away from the thing. When I speak of "appearances" I am speaking of something purely physical–the sort of thing, in fact, that, if it is visual, can be photographed. From a close-up photograph it is possible to infer a photograph of the same object at a distance, while the contrary inference is much more precarious. That is to say, there is a one-many relation between distant and close-up appearances. Therefore the distance appearance, regarded as a representation of the close-up appearance, is vague according to our definition. I think all vagueness in language and thought is essentially analogous to this vagueness which may exist in a photograph.

My own belief is that most of the problems of epistemology, in so far as they

are genuine, are really problems of physics and physiology; moreover, I believe that physiology is only a complicated branch of physics. The habit of treating knowledge as something mysterious and wonderful seems to me unfortunate. People do not say that a barometer "knows" when it is going to rain; but I doubt if there is any essential difference in this respect between the barometer and the meteorologist who observes it. *There is only one philosophical theory which seems to me in a position to ignore physics, and this is solipsism. If you are willing to believe that nothing exists except what you directly experience, no other person can prove that you are wrong, and probably no valid arguments exist against your view.* But if you are going to allow any inferences from what you directly experience to other entities, then physics supplies the safest form of such inferences. And I believe that (apart from illegitimate problems derived from misunderstood symbolism) physics, in its modern forms, supplies materials for answers to all philosophical problems that are capable of being answered, except the one problem raised by solipsism, namely: Is there any valid inference ever from an entity experienced to one inferred? On this problem, I see no refutation of the skeptical position. But the skeptical philosophy is so short as to be uninteresting; therefore it is natural for a person who has learnt to philosophize to work out other alternatives, even if there is no very good ground for regarding them as preferable.

Bibliography

[1] Aaby, Anthony A. *Lecture Notes on the SWEBOK* 2002.

[2] ACM *ACM Interactions* ACM

[3] ACM *ACM Queue* Vol. 1 No. 6 (September 2003)

[4] ACM Transactions on Information and System Security

[5] Agans, David J. *The 9 Indispensable Rules for Finding Evan the Most Elusive Software and Hardware Problems* AMACOM 2002.

[6] Atluri. *Security of Data and Transaction Processing*, Kluwer Academic Publications, 2000.

[7] Beck, Kent. *Extreme Programming Explained*

[8] The Berkeley/Stanford Recovery-Oriented Computing (ROC) Project `roc.cs.berkeley.edu`

[9] Bishop, Matt. *Robust Computing* CS Department, UC Davis 1998.

[10] Brooks, Frederick. Three Great Challenges for Half-Century-Old Computer Science. Journal of the ACM, Vol. 50, No. 1, January 2003, pp. 25-26.

[11] Brown et. al. *Anti Patterns: Refactoring Software, Architectures, and Projects in Crisis* John Wiley & Sons 1998.

[12] Buschmann et. al *Pattern-Oriented Software Architecture: A System of Patterns* Wiley 1996.

[13] Castano, Silvano, ed. *Database Security*. Addison-Wesley, 1994.

[14] Castro, M., Rodrigues, R., and Liskov, B. 2003. BASE: Using Abstraction to Improve Fault Tolerance. In *ACM TOCS*, Vol. 21., No. 3, August 2003, Pages 236-269.

[15] Cheswick, William R. and Steven M. Bellovin. *Firewalls and Internet Security*. Addison-Wesley, 1994.

[16] Christensen, Clayton M. “The rules of innovation” MIT Technology Review. June 2002. *The Innovator’s Dilemma* Harvard Business School Press. 1997.

[17] Coleman et. al. *Object-Oriented Development: the Fusion Method* Prentice-Hall.

[18] Cooper, James *Java design patterns: a tutorial* Addison-Wesley 2000.

[19] Conger, Sue A. and Mason, Richard O. *Planning and Designing Effective Web Sites* ITP 1998.

[20] Connell, Charles. Most Software Stinks. `www.chc-3/pub/beautifulsoftware.htm`

[21] Devlin, Keith. Why Universities Require Computer Science Students to Take Math. *Communications of the ACM*, Vol. 46. No. 9, September 2003, 37-39.

[22] Dijkstra, Edsger W. *A Discipline of Programming* Prentice-Hall 1976.

[23] Fowler, Martin *Refactoring: Improving the Design of Existing Code* Addison-Wesley 1999

[24] Fowler, Martin *Patterns of Enterprise Application Architecture* Addison-Wesley 2003.

[25] Gamma et. al *Design patterns: elements of reusable object-oriented software* Addison-Wesley 1995.

[26] Gardner, Tracy. *Inheritance relationships for disciplined software construction* Springer-Verlag, London, UK, 2002.

[27] T.R.G. Green and M. Petre, “Usability Analysis of Visual Programming Environments: a ’cognitive dimensions’ framework”, *J. Visual Languages and Computing*, vol.7 pp. 131-174, 1996.

[28] Gries, David. *The Sciene of Programming* Springer-Verlag 1981.

[29] Gurevich, Yuri 2000. Sequential Abstract State Machines Capture Sequential Algorithms, *ACM Transactions on Computational Logic*, Vol. 1, No. 1, July 2000, 77-111.

[30] Seth Hallem, David Park and Dawson Engler. Uprooting Software Defects at the Source. *ACM Queue* November 2003.

[31] Hallnäs, Lars and Redström, Johan. From Use to Presence: On the Expression and Aesthetics of Everyday Computational Things. *ACM Transactions on Computer-Human Interaction, Vol. 9*, No. 2, June 2002, Pages 106-124.

[32] Henderson, Austin. Design for What? Six Dimensions of Activity *ACM Interactions Vol VII.5* Sept & Oct 2000 pp. 17-22.

[33] Huggins, James K. and Wallace, Charles. "An Abstract State Machine Primer" *Computer Science Technical Report CS-TR-02-04* Michigan Technological University 2002.

[34] Instone, Keith. "Stress Test your Site", *Web Review* 1997. Also see `http://keith.instone.org/navstress`.

[35] IS-MCA. International Society for Mathematical and Computational Aesthetics `www.rci.rutgers.edu/~mleyton/ISMA.htm`

[36] Kühne, Thomas. *A Functional Pattern System for Object-Oriented Design* Verlag Dr. Kovac 1999.

[37] Lea, Doug *Christopher Alexander: An Introduction for Object-Oriented Designers* Software Engineering Notes Vol 19 No 1 Jan 1994.

[38] Lamport, Leslie 1989. A Simple Approach to Specifying Concurrent Systems *Communications of the ACM 32*, 1 (January 1989), 32-45.

[39] Lampson, Butler W. Hints for computer system design. *ACM Operating Systems Rev.* 15, 5 (Oct. 1983), pp33-48. Reprinted in *IEEE Software* 1, 1 (Jan. 1984), pp 11-28.

[40] Lee, Tae Eun (2002) *Architectural Design and Spiritual Life: Illustration and parallels.* 30th International Seminar on the Integration of Faith and Learning. Sahmyook University, Seoul Korea June 16-28, 2002.

[41] Michalewicz, Z. and Fogel, D. B. *How to solve it: Modern Heuristics* Springer-Verlag 2000.

[42] Moriconi, Mark and Qian, Xiaolei. "Correctness and Composition of Software Architecture" *Proceedings of ACM SIGSOFT'94: Symposium on Foundations of Software Engineering* 1994.

[43] Newell, A. and Simon, H. A. (1976). Computer Science as Empirical enquiry: Symbols and Search. *Communications of the ACM*, 19(3), pp. 113-126, March.

[44] Norman, Donald. *The Design of Everyday Things.* Doubleday, 1990.

[45] Polya, G. *How to Solve It*, 2nd ed., Princeton University Press, 1957.

[46] Raymond, Eric S. `catb.org/~esr/writings/taoup` *The Art of Unix Programming* Addison-Wesley 2003.

[47] Resenfeld and Morville's *Information Architecture for the World Wide Web.* O'Reilly, 2002.

[48] Schneier, Bruce. *Applied Cryptography.* John Wiley & Sons, 1996.

[49] Schneier, Bruce. *Secrets and Lies.* John Wiley & Sons, 2000.

[50] Shanks, Tansley, and Weber. Using Ontology to Validate Conceptual Models *Comunications of the ACM* Vol. 46, No. 10. October 2003.

[51] Shneiderman, Ben. *Designing the User Interface 3rd ed.*

[52] Shneiderman, Ben. "Creating Creativity: User Interfaces for Supporting Innovation" *ACM Transactions on Computer-Human Interaction*, Vol 7, No. 1, March 2000, Pages 114-138.

[53] Shneiderman, Ben. "Understanding Human Activities and Relationships" *Interactions* September & October 2002. pp. 40-53. - an excerpt from *Leonardo's Laptop: human needs and the New Computing Technologies* MIT Press 2002.

[54] Shneiderman, Ben. "Universal Usability with Multi-Layer Interface Design" *Proceedings of the 2003 Conference on Universal Usability* November 2003.

[55] Sklar, Joel. *Principles of Web Design* ITP 2000.

[56] Software Engineering Institute. `www.sei.cmu.edu/ata`

[57] Sommerville, Ian. *Software Engineering* 5th ed.

[58] Soni D. , Nord R. L., and Hofmeister C., "Software Architecture in Industrial Applications", Proceedings of the 17th International Conference on Software Engineering, April 1995.

[59] Spinellis, Diomidis. Reading, Writing, and Code. *ACM Queue* October 2003.

[60] Steinberg, Daniel H. and Palmer, Daniel W. *Extreme Software Engineering: a hands on approach* Pearson Education Inc 2004.

[61] SWEBOK Project. *Guide to the Software Engineering Body of Knowledge* Stone Man Trial Version 1.00 (May 2001)

[62] Theriault, Marlene L. and William Heney. *Oracle Security* O'Reilly and Associates, 1998.

[63] Tognazzini, Bruce. *First Principles of Interaction Design* `www.asktog.com/basics/firstPrinciples.html`

[64] URLs

- Security Forum - Security Design Patterns `www.opengroup.org/security/gsp.htm`

- Security Patterns.Org `www.securitypatterns.org`
- Common Front Group `cfg.cit.cornell.edu/cfg/design/contents.html`
- Tog. Interactive Design Solutions. `www.asktog.com`
- IBM Ease of Use `www.ibm.com/ibm/easy`
- Web Accessibility Initiative `www.w3.org/WAI`

[65] Viega, John and McGraw, Gary. *Building Secure Software* Addison-Wesley 2002.

[66] Welie.com `www.welie.com` Web and GUI design patterns